



## **AD/ADVANTAGE**

MANTIS Application Development Tutorial  
OS/390, VSE/ESA

P39-5026-01



---


# AD/Advantage® MANTIS Application Development Tutorial, OS/390, VSE/ESA

## Publication Number P39-5026-01

© 2001 Cincom Systems, Inc.  
All rights reserved

This document contains unpublished, confidential, and proprietary information of Cincom. No disclosure or use of any portion of the contents of these materials may be made without the express written consent of Cincom.

The following are trademarks, registered trademarks, or service marks of Cincom Systems, Inc.:

AD/Advantage®	/D CinDoc™	MANTIS®
C+A-RE™	/D CinDoc Web™	Socrates®
CINCOM®	/D Consulting™	Socrates® XML
Cincom Encompass®	/D Correspondence™	SPECTRA™
Cincom Smalltalk™	/D Correspondence Express™	SUPRA®
Cincom SupportWeb®	/D Environment™	SUPRA® Server
CINCOM SYSTEMS®	/D Solutions™	Visual Smalltalk®
	intelligent Document	VisualWorks®
gOOi™	Solutions™	
	Intermax™	

All other trademarks are trademarks or registered trademarks of:

Acucobol, Inc.	Micro Focus, Inc.
AT&T	Microsoft Corporation
Compaq Computer Corporation	Systems Center, Inc.
Data General Corporation	TechGnosis International, Inc.
Gupta Technologies, Inc.	The Open Group
International Business Machines Corporation	UNIX System Laboratories, Inc.
JSB Computer Systems Ltd.	

or of their respective companies.

Cincom Systems, Inc.  
55 Merchant Street  
Cincinnati, Ohio 45246-3732  
U. S. A.

PHONE: (513) 612-2300  
FAX: (513) 612-2000  
WORLD WIDE WEB: <http://www.cincom.com>

---

### Attention:

Some Cincom products, programs, or services referred to in this publication may not be available in all countries in which Cincom does business. Additionally, some Cincom products, programs, or services may not be available for all operating systems or all product releases. Contact your Cincom representative to be certain the items are available to you.

---

---

## Release information for this manual

*AD/Advantage MANTIS Application Development Tutorial, OS/390, VSE/ESA, P39-5026-01*, is dated October 30, 2001. This document supports Release 5.5.01 of MANTIS.

### We welcome your comments

We encourage critiques concerning the technical content and organization of this manual. At your convenience, please take the [survey](#) provided with the online documentation.

### Cincom Technical Support for AD/Advantage

<i>All customers</i>	Web:	<a href="http://supportweb.cincom.com">http://supportweb.cincom.com</a>
<i>U. S. A. customers</i>	Phone	1-800-727-3525
	:	
	FAX:	(513) 612-2000
		Attn: AD/Advantage Support
	Mail:	Cincom Systems, Inc.
		Attn: AD/Advantage Support
		55 Merchant Street
		Cincinnati, OH 45246-3732
		U. S. A.
<i>Customers outside U. S. A.</i>	All:	Visit the support links at <a href="http://www.cincom.com">http://www.cincom.com</a> to find contact information for your nearest Customer Service Center.

---



# Contents

<b>About this book</b>	<b>xi</b>
Using this document .....	xi
Document organization .....	xii
Conventions.....	xiv
MANTIS documentation series.....	xvii
Educational material .....	xviii
<b>Introduction</b>	<b>19</b>
Before you begin .....	20
Signing on to MANTIS.....	21
Signing off MANTIS.....	23
Introduction to the Burrys scenario.....	24
<b>Creating a screen</b>	<b>27</b>
Learning outline .....	27
Basic concepts: Understanding screen design .....	28
Using the screen design work area .....	28
Using screen design commands .....	32
Understanding screen fields.....	34
Step-by-step: Creating the customer browse screen .....	35
Step 1: Selecting the Screen Design Facility .....	35
Step 2: Creating and formatting screen fields .....	37
Step 3: Using the screen design library functions .....	42
Step 4: Updating field specifications .....	46
Step 5: Listing field specifications .....	53
Step 6: Listing repeat specifications.....	55
Step 7: Displaying the completed screen design .....	56
Step 8: Saving the screen .....	57
Step 9: Viewing the directory of screens.....	58
Step 10: Printing the completed screen design.....	59
Step 11: Using screen design window mode .....	60

Exercises.....	76
Exercise 1: Creating the customer menu screen.....	77
Exercise 2: Creating the new customer entry screen .....	79
Exercise 3. Creating the state code entry screen .....	80
Exercise 4: Viewing the directory of screens.....	81

**Creating a MANTIS file 83**

Learning outline .....	83
Basic concepts: Understanding MANTIS file design .....	84
Step-by-step: Creating the customer information file .....	85
Step 1: Creating the file .....	86
Step 2: Updating the record layout .....	88
Step 3: Saving the file .....	93
Step 4: Viewing the directory of files.....	94
Step 5: Printing the completed file design .....	94
Exercise .....	95

**Creating a prompter 97**

Learning outline .....	97
Basic concepts: Understanding prompter design .....	98
Step-by-step: Creating the state codes prompter.....	99
Step 1: Setting the tabs for prompter design .....	100
Step 2: Creating the prompter .....	103
Step 3: Saving the prompter .....	106
Step 4: Viewing the directory of prompters.....	108
Step 5: Displaying the completed prompter design .....	109
Step 6: Printing the completed prompter design .....	110
Exercise .....	110

**Understanding MANTIS programming fundamentals 111**

Learning outline .....	111
Basic concepts: Understanding MANTIS language conventions .....	112
MANTIS number set .....	112
MANTIS character set.....	113
Understanding literals, symbolic names, variables, and expressions .....	113
Using symbolic names .....	115
Using literals, variables, and expressions.....	117
Step-by-step: Creating the customer menu program .....	120
Step 1: Using the EDIT Program Entry screen.....	121
Step 2: Entering a program in the Full-Screen Editor (FSE) .....	123
Step 3: Terminating the Full-Screen Editor .....	126

---

Exercises .....	127
Exercise 1: Verifying the menu program .....	127
Exercise 2: Adding records to the state codes file .....	128
Exercise 3: Adding records to the customer information file.....	129
<b>Using MANTIS programming statements and commands</b> .....	<b>131</b>
Learning outline .....	131
Basic concepts: Understanding statements and commands .....	132
Using MANTIS program statements .....	133
Including comments in a program .....	137
Using MANTIS programming commands.....	138
Step-by-step: Completing the menu program .....	142
Step 1: Adding a comment.....	143
Step 2: Finding and changing a program line .....	144
Step 3: Understanding the CHAIN statement .....	145
Step 4: Renumbering your program.....	146
Step 5: Saving your changes .....	146
Exercises .....	147
Exercise 1: Creating a stub for the customer entry program .....	147
Exercise 2: Creating a stub for the browse program.....	148
Exercise 3: Running the menu program.....	148
<b>Creating a browse program</b> .....	<b>149</b>
Learning outline .....	149
Basic concepts: Understanding MANTIS file access .....	150
Step-by-step: Writing a browse program.....	153
Step 1: Specifying a MANTIS internal file .....	154
Step 2: Retrieving the first record from the file.....	155
Step 3: Accessing help.....	157
Step 4: Using the LEVEL parameter with the FILE statement.....	158
Step 5: Clearing the records on the browse screen.....	159
Step 6: Adding a counter to the program .....	160
Step 7: Reading the file record-by-record .....	161
Step 8: Terminating the browse .....	162
Step 9: Sequencing and replacing the browse program .....	163
Exercises .....	164
Exercise 1: Writing a program to browse the state codes file .....	164
Exercise 2: Enhancing the state codes browse program.....	164

<b>Creating a data entry program</b>	<b>165</b>
Learning outline .....	165
Basic concepts: Understanding program control basics.....	166
Program control capabilities .....	166
Using the DO statement.....	167
Step-by-step: Creating the customer entry program .....	170
Step 1: Defining the file.....	171
Step 2: Inserting a record into the file .....	172
Step 3: Clearing the entry .....	172
Step 4: Calling the edit subroutine.....	173
Step 5: Using the IF-END structure to test for errors.....	174
Step 6: Testing your program structure .....	175
Step 7: Adding an edit routine to test for an entry in a field.....	176
Step 8: Adding an edit routine to test the number of characters entered...	181
Step 9: Adding an edit routine to validate the state code .....	185
Step 10: Displaying the state codes prompter .....	191
Step 11: Adding an edit routine to test for a duplicate entry.....	195
Step 12: Enhancing the customer entry program .....	200
Exercise .....	204
 <b>Using component engineering</b>	 <b>205</b>
Learning outline .....	205
Basic concepts: Understanding component engineering .....	206
Designing components .....	208
Creating source programs .....	209
Establishing naming conventions for source programs .....	211
Accessing component engineering options .....	212
Step-by-step: Applying component engineering to the Burrys programs .....	214
Step 1: Defining the components.....	215
Step 2: Coding and nominating the SOURCE statement.....	218
Step 3: Coding the REPLACE statement .....	219
Step 4: Decomposing the program .....	220
Step 5: Viewing the program list .....	222
Step 6: Adding a component to the customer browse program .....	223
Step 7: Cross-referencing the source program .....	228
Step 8: Viewing the Bill of Materials and Component Where Used lists ....	230
Exercise .....	233
 <b>What's next?</b>	 <b>235</b>



<b>Exercise examples</b>	<b>237</b>
Chapter 1: Introduction.....	237
Chapter 2: Creating a screen .....	237
Exercise 1: Creating the customer accounts menu.....	238
Exercise 2: Creating the new customer entry screen .....	239
Exercise 3: Creating the state code entry screen .....	240
Exercise 4: Viewing the directory of screens .....	241
Chapter 3: Creating a MANTIS file.....	242
Chapter 4: Creating a prompter.....	242
Chapter 5: Programming fundamentals .....	243
Exercise 1: Verifying the menu program .....	243
Exercise 2: Adding records to the state codes file .....	244
Exercise 3: Adding records to the customer information file.....	244
Chapter 6: Using the Full-Screen Editor .....	245
Exercise 1: Creating a stub for the customer entry program .....	245
Exercise 2: Creating a stub for the browse program.....	245
Chapter 7: Creating a browse program.....	246
Exercise 1: Writing a program to browse the state codes file .....	246
Exercise 2: Enhancing the state codes browse program.....	247
Chapter 8: Creating a data entry program .....	248
Chapter 9: Using component engineering.....	250
Chapter 10: What's next? .....	251
 <b>Index</b>	 <b>253</b>



# About this book

---

---

## Using this document

This tutorial provides you with a basic introduction to designing MANTIS screens, files, and prompters, and to creating programs using the MANTIS language. For more detailed information on these topics, refer to *MANTIS Facilities, OS/390, VSE/ESA*, P39-5001 and *MANTIS Program Design and Editing, OS/390, VSE/ESA*, P39-5013.

This manual is not intended to teach you how to program, but rather to show you the basics of MANTIS so you can apply them to your data processing environment. Exercises appear at the end of each chapter. Take your time with the lessons and follow directions carefully.

This manual will guide you through most of the steps necessary to create an application in MANTIS. At each step, you will use MANTIS to create part of the application, and the exercises will teach you about MANTIS and its capabilities. Upon completion, you should be ready to begin writing or maintaining your own MANTIS applications.

Many of the MANTIS programming statements and commands appear in this tutorial, along with their syntax. “Conventions” on page xiv describes the conventions that are used in the syntax notation. Read this section carefully before you begin the programming lessons in this tutorial. For more detailed information on programming statements and commands, refer to *MANTIS Language, OS/390, VSE/ESA*, P39-5002.

## Document organization

The information in this manual is organized as follows:

### Chapter 1—Introduction

Provides a brief description of MANTIS, fundamental rules for its use (including sign-on and sign-off instructions), and an introduction to the Burrys application scenario that is used as the basis for this tutorial.

### Chapter 2—Creating a screen

Describes the screen design process and shows you how to design the first screen in the Burrys scenario. The other three screens required for the Burrys application are presented as exercises.

### Chapter 3—Creating a MANTIS file

Describes the file design process and shows you how to design the first file of the Burrys scenario. The other file required for the Burrys application is presented as an exercise.

### Chapter 4—Creating a prompter

Describes the prompter design process and shows you how to design the prompter required for the Burrys scenario.

### Chapter 5—Understanding MANTIS programming fundamentals

Explains MANTIS language conventions and shows you how to enter a program in the Full-Screen Editor. The exercises show you how to run a program to insert records into a file.

### Chapter 6—Using MANTIS programming statements and commands

Describes the MANTIS programming environment in more detail, explains basic MANTIS programming statements, and shows you how to write the menu program for the Burrys application. In the exercises, you create the stubs for the browse and customer entry programs.

### Chapter 7—Creating a browse program

Describes how to complete a program that displays the contents of the Burrys customer file. In the exercises, you write a program to browse the state codes file.

### Chapter 8—Creating a data entry program

Shows you how to complete a program that validates and inserts new customer records into the customer file. In the exercise, you design a maintenance program that reads particular records from the Burrys customer file and deletes or updates them.

### **Chapter 9—Using component engineering**

Shows you how to apply component engineering to create reusable building blocks of MANTIS code that simplify program development and maintenance.

### **Chapter 10—What's next?**

Summarizes the tutorial and provides references to further information on topics of interest.

### **Appendix A—Exercise examples**

Provides examples of solutions for the exercises in this tutorial.

### **Index**

# Conventions

The following table describes the conventions used in this document series:

Convention	Description	Example
Constant width type	Represents screen images and segments of code.	Screen Design Facility GET NAME LAST INSERT ADDRESS
Yellow-highlighted, red code or screen text	Indicates an emphasized section of code or portion of a screen.	00010 ENTRY COMPOUND 00020 .SHOW"WHAT IS THE CAPITAL AMOUNT?" 00030 .OBTAIN INVESTMENT 00040 EXIT
Slashed b (b)	Indicates a space (blank).  The example indicates that a password can have a trailing blank.	WRITEPASSb
Brackets [ ]	Indicate optional selection of parameters. (Do not attempt to enter brackets or to stack parameters.) Brackets indicate one of the following situations.  A single item enclosed by brackets indicates that the item is optional and can be omitted.  The example indicates that you can optionally enter a program name.  Stacked items enclosed by brackets represent optional alternatives, one of which can be selected.  The example indicates that you can optionally enter NEXT, PRIOR, FIRST, or LAST. (NEXT is underlined to indicate that it is the default.)	COMPOSE [program-name]  <div><u>NEXT</u> PRIOR FIRST LAST</div>

Convention	Description	Example
Braces { }	<p>Indicate selection of parameters. (Do not attempt to enter braces or to stack parameters.) Braces surrounding stacked items represent alternatives, one of which you must select.</p> <p>The example indicates that you must enter FIRST, LAST, or a value for <i>begin</i>.</p>	<pre> {FIRST   begin LAST} </pre>
<u>Underlining</u> (In syntax)	<p>Indicates the default value supplied when you omit a parameter.</p> <p>The example indicates that if you do not specify ON, OFF, or a row and column destination, the system defaults to ON.</p>	<pre> SCROLL [ON         OFF         [row] [, col]] </pre>
	<p>Underlining also indicates an allowable abbreviation or the shortest truncation allowed.</p> <p>The example indicates that you can enter either PRO or PROTECTED.</p>	<pre> PROTECTED </pre>
Ellipsis points...	<p>Indicate that the preceding item can be repeated.</p> <p>The example indicates that you can enter (A), (A,B), (A,B,C), or some other argument in the same pattern.</p>	<pre> (argument,...) </pre>

Convention	Description	Example
UPPERCASE	Indicates MANTIS reserved words. You must enter them exactly as they appear.  The example indicates that you must enter CONVERSE exactly as it appears.	CONVERSE <i>name</i>
<i>Italics</i>	Indicate variables you replace with a value, a column name, a file name, and so on.  The example indicates that you can supply a name for the program.	COMPOSE [ <i>program-name</i> ]
Punctuation marks	Indicate required syntax that you must code exactly as presented.  (    )    parentheses . ,        comma : ; '        single quotation mark "    "    double quotation marks	[LET] <sub>v</sub> $\begin{bmatrix} (i) \\ (i,j) \end{bmatrix}$ [ROUNDED( <i>n</i> )] = <i>e1</i> [ <i>e2,e3...</i> ]



---

## MANTIS documentation series

MANTIS is an application development system designed to increase productivity in all areas of application development, from initial design through production and maintenance. MANTIS is part of AD/Advantage, which offers additional tools for application development. Listed below are the manuals offered with MANTIS in the IBM® mainframe environment, organized by task. You may not have all the manuals listed here.

### MASTER User tasks

- ◆ *MANTIS Installation, Startup, and Configuration, MVS/ESA, OS/390, P39-5018*
- ◆ *MANTIS Installation, Startup, and Configuration, VSE/ESA, P39-5019*
- ◆ *MANTIS Administration, OS/390, VSE/ESA, P39-5005*
- ◆ *MANTIS Messages and Codes, OS/390, VSE/ESA, P39-5004\**
- ◆ *MANTIS Administration Tutorial, OS/390, VSE/ESA, P39-5027*
- ◆ *MANTIS XREF Administration, OS/390, VSE/ESA, P39-0012*

### General use

- ◆ *MANTIS Quick Reference, OS/390, VSE/ESA, P39-5003*
- ◆ *MANTIS Facilities, OS/390, VSE/ESA, P39-5001*
- ◆ *MANTIS Language, OS/390, VSE/ESA, P39-5002*
- ◆ *MANTIS Program Design and Editing, OS/390, VSE/ESA, P39-5013*
- ◆ *MANTIS Messages and Codes, OS/390, VSE/ESA, P39-5004\**
- ◆ *AD/Advantage Programming, P39-7001*
- ◆ *MANTIS DB2 Programming, OS/390, VSE/ESA, P39-5028*

- ◆ *MANTIS SUPRA SQL Programming, OS/390, VSE/ESA*, P39-3105
- ◆ *MANTIS XREF, OS/390, VSE/ESA, OpenVMS*, P39-0011
- ◆ *MANTIS Entity Transformers*, P39-0013
- ◆ *MANTIS DL/I Programming, OS/390, VSE/ESA*, P39-5008
- ◆ *MANTIS SAP Facility, OS/390, VSE/ESA*, P39-7000
- ◆ *MANTIS WebSphere MQ Programming*, P39-1365
- ◆ *MANTIS Application Development Tutorial, OS/390, VSE/ESA*, P39-5026



---

Manuals marked with an asterisk (\*) are listed twice because you use them for both MASTER User tasks and general use tasks.

---

## Educational material

AD/Advantage and MANTIS educational material is available from your regional Cincom education department.

# 1

## Introduction

MANTIS is a comprehensive application development system designed to increase productivity in all areas of application development—from initial design through production.

MANTIS offers design facilities, prototyping capabilities, testing and debugging tools, and an advanced, high-level programming language. Among other things, MANTIS enables you to:

- ◆ Design and create screens that display data attractively by taking full advantage of large screen capabilities and the features available on today's terminals (such as color, reverse video, blinking, etc.).
- ◆ Design and create permanent files for data storage and manipulation.
- ◆ Create and test programs interactively using structured programming concepts.

All MANTIS facilities are completely interactive. This means that once a program, screen, or file is created, it is immediately available for display and review by end users. This eliminates the need for precompiling, compiling, binding, coding Job Control Language, and other activities normally associated with application development.

This tutorial will show you how to use the MANTIS screen, file, and prompter design facilities. It will also demonstrate how you can use the MANTIS programming language and the Program Design Facility to quickly and easily build programs that meet your application development needs.

## Before you begin

Each MANTIS installation has a person (or persons) designated as the MANTIS Master User. The Master User has access to certain facilities and information not available to all MANTIS users.

You will normally access MANTIS from a menu program that already exists at your site. Check with your Master User for instructions. Your Master User will also supply you with a valid user ID and password to sign on to MANTIS.



The following screen illustration shows the standard Facility Selection menu that is provided with MANTIS:

FAC002	MANTIS Facility Selection Menu	YYYY:MM:DD																																								
	BURRYS	HH:MM:SS																																								
Please select one of the menu options below.																																										
<table border="0"> <tr> <td>Run a Program by Name .....</td> <td>1</td> <td>Sign On as Another User ....</td> <td>11</td> </tr> <tr> <td>Display a Prompter .....</td> <td>2</td> <td>Search Facility .....</td> <td>12</td> </tr> <tr> <td>Design a Program .....</td> <td>3</td> <td>Query Report Writer .....</td> <td>13</td> </tr> <tr> <td>Design a Screen .....</td> <td>4</td> <td>Directory Facility .....</td> <td>14</td> </tr> <tr> <td>Design a MANTIS File View ..</td> <td>5</td> <td>Transfer Facility .....</td> <td>15</td> </tr> <tr> <td>Design a Prompter .....</td> <td>6</td> <td>Cross Reference Facility ...</td> <td>16</td> </tr> <tr> <td>Design an Interface .....</td> <td>7</td> <td>Entity Transformers .....</td> <td>17</td> </tr> <tr> <td>Design a TOTAL File View ...</td> <td>8</td> <td>Universal Export Facility ..</td> <td>18</td> </tr> <tr> <td>Design an External File View</td> <td>9</td> <td>Print Facility .....</td> <td>19</td> </tr> <tr> <td>DL/I Access View .....</td> <td>10</td> <td></td> <td></td> </tr> </table>			Run a Program by Name .....	1	Sign On as Another User ....	11	Display a Prompter .....	2	Search Facility .....	12	Design a Program .....	3	Query Report Writer .....	13	Design a Screen .....	4	Directory Facility .....	14	Design a MANTIS File View ..	5	Transfer Facility .....	15	Design a Prompter .....	6	Cross Reference Facility ...	16	Design an Interface .....	7	Entity Transformers .....	17	Design a TOTAL File View ...	8	Universal Export Facility ..	18	Design an External File View	9	Print Facility .....	19	DL/I Access View .....	10		
Run a Program by Name .....	1	Sign On as Another User ....	11																																							
Display a Prompter .....	2	Search Facility .....	12																																							
Design a Program .....	3	Query Report Writer .....	13																																							
Design a Screen .....	4	Directory Facility .....	14																																							
Design a MANTIS File View ..	5	Transfer Facility .....	15																																							
Design a Prompter .....	6	Cross Reference Facility ...	16																																							
Design an Interface .....	7	Entity Transformers .....	17																																							
Design a TOTAL File View ...	8	Universal Export Facility ..	18																																							
Design an External File View	9	Print Facility .....	19																																							
DL/I Access View .....	10																																									
F1=HELP F3=END F12=CANCEL																																										

Your user ID name displays directly below the MANTIS Facility Selection Menu heading. (The illustrations throughout this tutorial show BURRYS as the user ID and BURRYS as its password. If your own user ID and password are different, use them instead.)

The Facility Selection menu lists the facilities that are available for your MANTIS installation. Your Master User may have omitted some of these facilities, and/or added new facilities to meet your specific needs.

The cursor appears in the action field. To access a facility from the menu, enter the number of the facility in the action field, and press ENTER.

---

## Signing off MANTIS

To sign off, press the CANCEL key until you reach the Facility Selection menu. Your Master User will provide you with directions for signing off MANTIS from this point.



---

On a 3270 terminal, the PA2 key is equivalent to the CANCEL key, which returns you to a higher screen. However, the CANCEL key may be different for your installation. Check with your Master User if you are not sure which key to use as the CANCEL key.

---



---

You can sign off at any time during the tutorial session. But, before you do, be sure to save or replace all design work through the library functions in the facility where you are currently working.

---

If you forget to save your work, you will receive a message “UNSAVED CHANGES EXIST – USE CANCEL TO CONTINUE WITH TERMINATION.”

---

When you are ready to begin again, use the start-up procedure in “[Signing on to MANTIS](#)” on page 21. Then, if you are continuing work on an existing design, use the library functions of the corresponding design facility to fetch the design into your work area.

## Introduction to the Burrys scenario

---

In this tutorial, you will use the MANTIS facilities to create an application for a fictitious corporation named “Burrys,” an electrical goods retailer with branches in various parts of the country.

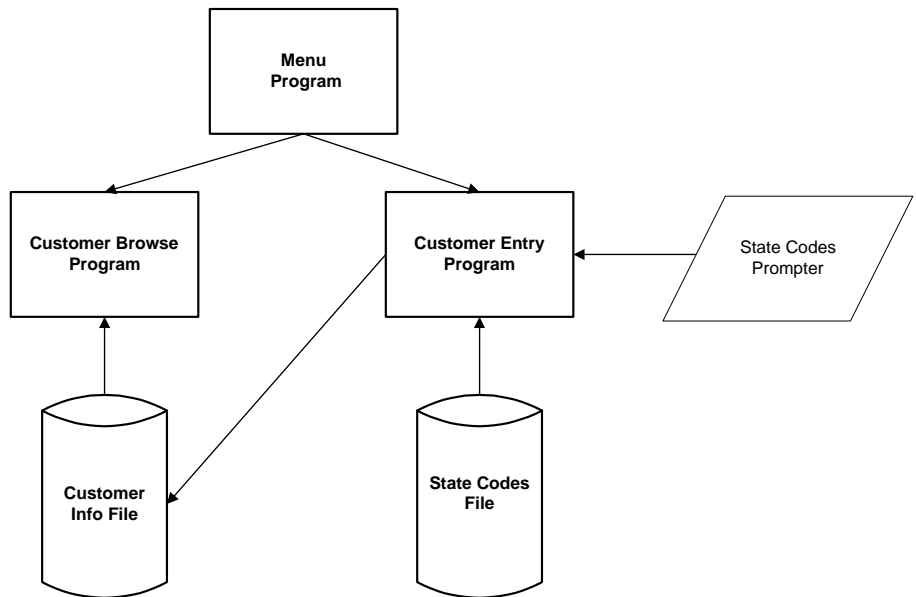
Managers at each branch are currently unable to establish credit for new customers without first contacting the corporate accounting department. The accounting department establishes a customer’s credit rating and credit limit, and assigns a customer number.

Over the past two years, however, Burrys has increased its sales by 135% and its customer base by 53%. Branch managers all agree that the present system for customer file maintenance and inquiry is inadequate.

The branch managers have requested, and received approval for, an online, automated system that will allow them to access the corporate database. The system will enable managers to request customer information and add new customers to the database.



The following diagram shows the Burrys customer accounts system:

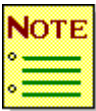


Burrys has chosen MANTIS as its application development system. The prototyping capability and ease of use of MANTIS will minimize the time and cost of developing and implementing the system.

In the following chapters, you will develop the Burrys customer accounts system. The system will require:

- ◆ Four screens:
  - Menu (CUST\_MENU)
  - Customer entry (CUST\_ENTRY)
  - Customer browse (CUST\_BROWSE)
  - State code entry (STATE\_CODE)
- ◆ Two files:
  - Customer information (CUST\_INFO)
  - State codes (STATE\_CODES)
- ◆ A state codes prompter (STATE\_CODES)
- ◆ Three programs:
  - Menu (CUST\_MENU)—Displays the menu screen and allows the user to choose the customer entry and browse programs, or exit from the Burrys system.
  - Customer entry (CUST\_ENTRY)—Validates the information provided about a customer and inserts the data into the customer information file.
  - Customer browse (CUST\_BROWSE)—Displays the contents of the customer information file on the customer browse screen.

In the Design Facilities portion of this tutorial (chapters 2 through 4), you will create the screens, files, and prompter. Once you have completed these items, you will use the Program Design Facility and Full-Screen Editor to create and run the menu, customer entry, and customer browse programs.



---

The examples in this tutorial show MANTIS keys that are standard for a 3270 terminal. Certain key assignments may differ for your environment, depending on how your Master User has configured MANTIS, and upon the terminal emulator you are using and how it has been configured.

---

# 2

## Creating a screen

Before you begin creating the Burrys application, you'll need to know some basic features of the MANTIS Screen Design Facility. This chapter introduces you to these concepts, then guides you step-by-step through the process of creating the customer browse screen for the Burrys customer accounts application. The other three screens required for the application are presented as exercises.

### Learning outline

In this chapter, you will learn how to perform the following:

- ◆ Use the screen design work area.
- ◆ Access the Screen Design Facility menu and select options on it.
- ◆ Design a screen by creating and formatting headings and data fields.
- ◆ Save a new screen, or changes to an existing screen.
- ◆ Specify the data type and other attributes for a screen field.
- ◆ View the attributes that you have specified for screen fields.
- ◆ Assign repeat specifications to screen fields.
- ◆ Display a completed screen design.
- ◆ View the directory of screens.
- ◆ Print the completed screen design.
- ◆ Use window mode to create and view screens that are larger than the physical display.

## Basic concepts: Understanding screen design

Before you begin designing your screens for the Burrys application, let's take a look at some basic concepts used in the MANTIS screen design process.

### Using the screen design work area

MANTIS provides a screen design work area that is 255 rows by 255 columns.

A *domain* is the space occupied by a defined screen or field. Domains ensure that the screen and field definitions (including attributes and repeat specifications) you make during screen design are retained until you modify them. The Screen Design Facility has two types of domains:

- ◆ **Screen domain.** The size of your screen design. This domain is called the *logical screen* to distinguish it from the physical screen (your terminal or emulator).
- ◆ **Field domain.** An invisible area on the screen that begins with the byte preceding a field and extends to the end of that field.

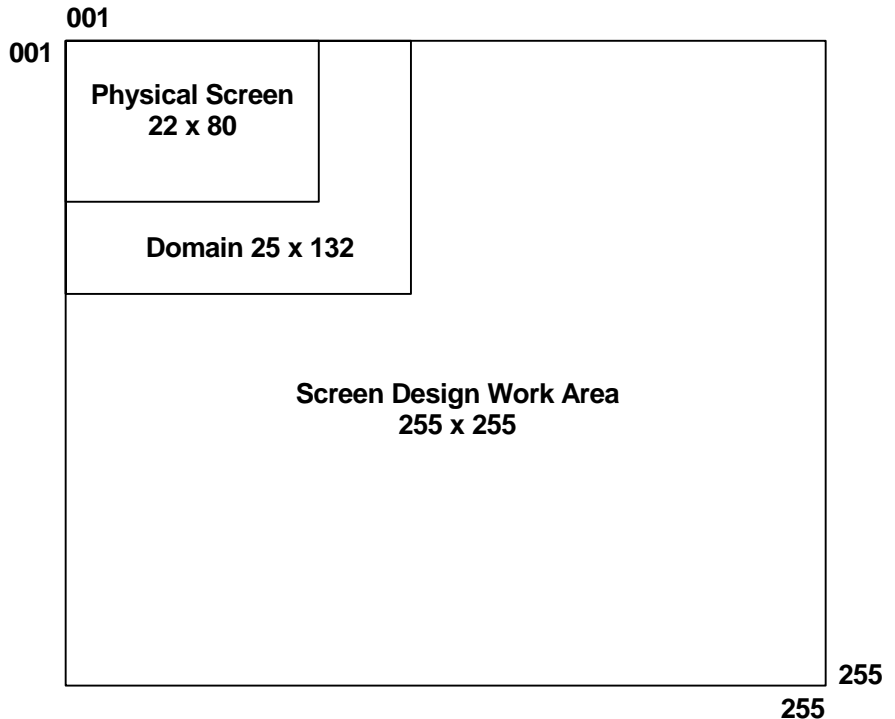


---

The screen design example in this chapter assumes that you are using a 3278 Model 2 terminal, which is non-color and has a display area of 24 lines x 80 columns. The attributes (such as the intensity of a field) may yield different results on your terminal or emulator.

---

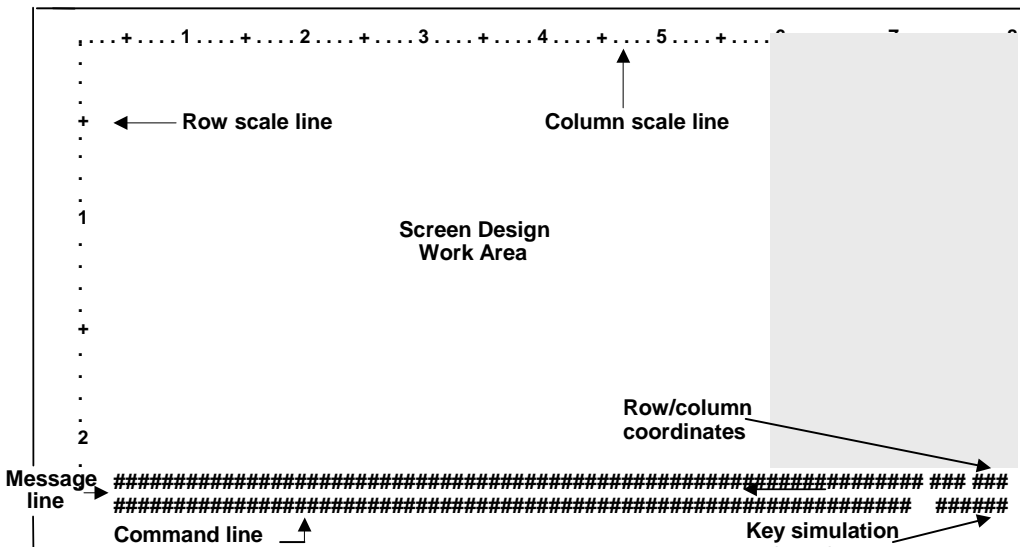
Each screen that you create has a domain within the work area. For example, in this illustration the screen domain is 25 rows long and 132 columns wide:



The first set of Burrys screens that you design will use a 22x80 screen domain, so the logical and physical screen sizes are the same for a Model 2 terminal.

If the row scale line is turned off during screen design, MANTIS reserves the first column of the screen. If the row scale line is turned on, MANTIS reserves the first three columns of the screen. The columns that MANTIS reserves are not part of the screen design work area.

The last two physical lines of the screen design work area are reserved for the message, row/column coordinates (not displayed during the screen design painting session), command line, and key simulation field. These fields remain on your screen even when you move your physical screen around the work area:



Blank spaces fill the displayed screen up to the last 20 columns. The last 20 columns (shown in gray on the previous screen) are empty; that is, they contain nulls. This allows you the flexibility of inserting fields in your design and shifting the remaining fields to the right. When you position data or heading fields in this null area, you must insert the required blanks using the space bar. (If you use the PF keys to copy or move a field into this area, you do not need to insert blanks.)

The following table describes the reserved fields on the screen design work area:

Field	Length	Description
Message line	Terminal width minus 10 columns	Displays system messages.
Row/column coordinates	3 bytes for each	Displays the current row and column coordinates of the upper-left corner of the screen design (when in window mode).
Command line	Terminal width less 8 columns	Provides a field for entering commands.
Key simulation field	6 bytes	Provides a field for entering function keys, if your keyboard does not have them.
Row scale line	3 bytes per line	Row scale ruler (when present). Always occurs on the left edge of the physical screen. Toggled on and off by PF9 or PF21.
Column scale line	Terminal width	Column scale ruler (when present). May occur on any line in the design area. Toggled on and off and positioned by PF3 or PF15.

The row and column scale lines (when displayed) indicate the current row and column position within the work area. These scale lines change to reflect your position as you move around the work area. They will not appear as part of your finished design, but are provided to assist you with your design layout.



Use PF3 or PF15 to display, move, and remove the column scale line.  
Use PF9 or PF21 to display and remove the row scale line.

In the Screen Design Facility, row and column coordinates appear automatically in the lower right corner of the screen only when you display a completed screen design that is outside the boundaries of the terminal (that is, the screen domain (logical terminal) is larger than the physical screen).

## Using screen design commands

You can use your terminal screen as a moveable window, scrolling around the screen design to view different sections. To do so, you use MANTIS screen design commands to move the window and edit your screen design.

You specify these commands by using PF keys, or by entering a command on the command line. (Commands that you enter on the command line are described on page 34.) The following table lists the available PF keys, and describes how they work:

PF key	Function	Description
PF1 or PF13	Insert a line	If you place the cursor on, or before, the first character in a line and then press PF1 or PF13, MANTIS shifts the entire line down one line. If you place the cursor after the first character in the line and then press PF1 or PF13, MANTIS inserts a blank line beginning at the end of the current field.
PF2 or PF14	Delete a line	If the cursor is in position 1 of the line, MANTIS deletes that line. Otherwise, MANTIS deletes all fields that start within 80 columns to the right of the cursor, including those fields outside the window boundary up to, but not including, the corresponding cursor position on the next line. Fields on following lines that are not deleted move up one line.
PF3 or PF15	Insert, move, or delete the column scale line	Place the cursor on the line in the screen design work area where you want the column scale line to appear; then, press PF3 or PF15. The column scale line displays, temporarily overlaying any data that was displayed on this line. To clear the column scale line from the screen, press PF3 or PF15 again. The data that was under the scale line reappears. You cannot place the column scale line on the last two lines of the physical screen.
PF4 or PF16	Move a field	Position the cursor anywhere in the field you want to move and press PF4 or PF16. Move the cursor to where the first character of the field should appear and press PF4 or PF16 again. The field, along with its attribute information, moves to the new position. If you try to move a field that is too long to fit within the 255-column limit, MANTIS moves the field, but truncates the overflow.



PF key	Function	Description
PF5 or PF17	Copy a field	Position the cursor anywhere in the field that you want to copy and press PF5 or PF17. Then, move the cursor to the position where the first character of the copied field should appear, and press PF5 or PF17 again. The field, along with its attribute information, is copied. If you try to copy a field that is longer than 255 columns, MANTIS copies the field, but truncates the overflow.
PF6 or PF18	Delete a field	Position the cursor anywhere in the field that you want to delete and press PF6 or PF18. MANTIS deletes only the field where the cursor is placed; other fields on the same line remain in the same location.
PF7 or PF19	Scroll screen up	Scrolls the screen up 10 lines. The column scale line maintains its physical display position on the screen.
PF8 or PF20	Scroll screen down	Scrolls the screen down 10 lines. The column scale line maintains its physical display position on the screen.
PF9 or PF21	Display or remove the row scale line	Displays the row scale line in column 2. The screen design moves two positions to the right. Press PF9 or PF21 again to remove the row scale line. The screen design moves left two positions.
PF10 or PF22	Scroll screen left	Scrolls the screen left 20 columns. The row scale line maintains its physical position on the screen.
PF11 or PF23	Scroll screen right	Scrolls the screen right 20 columns. The row scale line maintains its physical position on the screen.
PF12 or PF24	Return window to origin	Returns the window to the origin (row 1, column 1). Both row and column scale lines maintain their physical positions on the screen.

You can also perform editing functions by entering commands on the command line. The following table lists the Screen Design Facility commands, along with a detailed description of each (underlined letters indicate command abbreviations):

Command	Description
CLEAR	Clears the current screen design from the work area
<u>C</u> OPY	Copies a single line, or a range of lines
<u>D</u> EFAULTS	Displays the default row and column domains and allows you to update them
<u>D</u> ELETE	Deletes a single line, or a series of lines
HELP	Displays help prompts for PF keys and commands (a version of the information in this and the preceding table)
<u>I</u> NSERT	Inserts a single line, or a series of lines
<u>M</u> OVE	Moves a single line, or a series of lines
NEW	Clears the current screen design from the work area

## Understanding screen fields

You can specify two types of fields for a screen design:

- ◆ **Data fields.** These fields are designated by a *mask character* (usually hash signs (#), unless your Master User has configured MANTIS to use another mask character). Data appears in these fields as it is exchanged with a running program. When you design a screen, you assign symbolic names to the data fields and specify attributes for the fields (for example, bright intensity).
- ◆ **Heading fields.** These fields may contain any valid MANTIS character except the mask character. These fields always appear on the final display exactly as you enter them during screen design, with the exception of the blank fill character (in our example, the vertical bar).

## Step-by-step: Creating the customer browse screen

Now that you understand some basic concepts of screen design, you're ready to design your first MANTIS screen.

### Step 1: Selecting the Screen Design Facility

When you sign on to MANTIS, the MANTIS Facility Selection menu appears:

```

FAC002                                MANTIS Facility Selection Menu          YYYY:MM:DD
                                      BURRYS                               HH:MM:SS
Please select one of the menu options below.

_____
Run a Program by Name ..... 1  Sign On as Another User .... 11
Display a Prompter ..... 2  Search Facility ..... 12
Design a Program ..... 3  Query Report Writer ..... 13
Design a Screen ..... 4  Directory Facility ..... 14
Design a MANTIS File View .. 5  Transfer Facility ..... 15
Design a Prompter ..... 6  Cross Reference Facility ... 16
Design an Interface ..... 7  Entity Transformers ..... 17
Design a TOTAL File View ... 8  Universal Export Facility .. 18
Design an External File View 9  Print Facility ..... 19
DL/I Access View ..... 10

F1=HELP  F3=END  F12=CANCEL

```

Your user ID appears at the top of the screen, unless your Master User has changed this menu. (In this and all following examples, the user ID is BURRYS.) The cursor is positioned in the action field, where you indicate your facility selection.

You want to design a screen, so select the Design a Screen facility by entering 4 in the action field and pressing ENTER. The Screen Design Facility menu appears, and the cursor is positioned in the action field at the bottom of the screen:

```
ASP001                                M A N T I S

                                     Screen Design Facility

Create or update a screen ..... 1
Update field specifications ..... 2
List field specifications ..... 3
Update repeat specifications ..... 4
List repeat specifications ..... 5
Display completed design ..... 6
Library functions ..... 7
Directory of screens ..... 8
Print completed design ..... 9
Terminate this facility ..... CANCEL

                                     : _ :
```

Because you are designing a new screen, you will follow the sequence of functions as they are listed on the Screen Design Facility menu (that is, first create a screen, then update field specifications, etc.).



You can select a design facility menu option in one of two ways: enter the corresponding number in the action field and press ENTER, or press the corresponding PF key.

## Step 2: Creating and formatting screen fields

In this step, you will create and format all first occurrences of your screen fields. To begin, select the Create or update a screen option by entering 1 in the action field and pressing ENTER, or by pressing PF1. Then, without moving the cursor, press PF3 or PF15 to display the column scale line, as shown in the following screen:

```

. ....+....1....+....2....+....3....+....4....+....5....+....6....+....7....+...
.
.
.
+
.
.
.
.
1
.
.
.
.
+
.
.
.
.
2
.
.
ASP026A: 'ENTER' to update, HELP for instructions, CANCEL to terminate

```

Instructions on the last line of the work area provide help and exit information. The remaining space is available for your screen design.



Remember that your screen design may be as large as 255 rows by 255 columns. You can scroll around the work area by pressing PF7 (up), PF8 (down), PF10 (left), or PF11 (right).

The cursor is positioned in the fourth column of the screen because MANTIS reserves the first column and the row scale line occupies two columns. If you try to enter data in these columns, MANTIS will freeze the screen. If that happens, press RESET, move the cursor to the fourth column, and continue with your design.

For this screen, you will define both heading and data fields. Remember, heading fields appear in the final display exactly as you enter them in screen design, except for the blank fill character (in this case, the vertical bar).

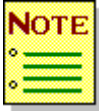
Place the cursor anywhere on line three and press PF3 or PF15 to reposition the column scale line. Enter the heading fields as shown below, using a vertical bar to separate letters and words as indicated:

[illegible]

If you make an error while entering your design, back space and type over the mistake with correct data, or erase the incorrect data with the space bar. To undo all the changes since the last time you pressed ENTER or a PF key, press the CLEAR key.

Because MANTIS interprets a blank ( ) as designating a new field, you should enter a blank fill character (in this case, the vertical bar character) between the words and letters of heading fields. The blank fill character connects words and letters into one field so that data is transmitted more efficiently. (For example, the heading BUURRYS assumes five heading fields, but B U R R YS assumes only one heading field.) When the screen is displayed, the vertical bar does not appear. (The blank fill character is changeable on the library functions screen, so that you can display the vertical bar if you wish.)

When you finish entering the heading fields, press ENTER. MANTIS checks your entry for errors and temporarily stores the design in your work area.



MANTIS may display an error message if an error is made at any point in the screen design process. For more information on messages, refer to the *MANTIS Messages and Codes, OS/390, VSE/ESA*, P39-5004.

Using the information in the table below the screen, add two lines of data fields (designated by hash signs):

```
.
.                                     B|U|R|Y|S
.                                CUSTOMER|BROWSE
. ....+....1.....+....2.....+....3.....+....4.....+....5.....+....6.....+....7.....+....
CUSTOMER||| ||| |CUSTOMER||| ||| |BRANCH||| ||| |CREDIT||| ||| |CREDIT
+   NUMBER||| ||| |||NAME||| ||| |||NUMBER||| ||| |RATING||| ||| |||LIMIT
.
. #####
.
.
.
1
.
.
.
+
.
.
.
.
2
. #####
ASP026A: 'ENTER' to update, HELP for instructions, CANCEL to terminate
```

Field	Field length
CUSTOMER NUMBER	6
CUSTOMER NAME	20
BRANCH NUMBER	4
CREDIT RATING	2
CREDIT LIMIT	6 (includes the edit character "\$")
MESSAGE	76

Notice that instead of using vertical bars between fields as you did with heading fields, you now enter blank spaces after each data field.

Note two other points about the preceding screen:

- ◆ You are entering an edit character (\$) under CREDIT LIMIT. You may use any valid MANTIS character as an edit character except the mask character (#, or some other character designated on the library functions screen). You may also use any character that you can enter from the keyboard (for example, national characters, punctuation, etc.)

Edit characters display only when there is data in the field. (Edit characters are used by numeric data types only. Information on text vs. numeric data fields will be explained in chapter 5 of this tutorial.)

- ◆ You are designating a data field for messages across the bottom of the screen. When the screen is used in a program, this field can display help messages, policy reminders, or any other information a user might need to interpret the screen.

When you have finished entering the data fields, press ENTER.



You have now defined all the heading fields and the first occurrences of the data fields for this screen. Press the CANCEL key to return to the Screen Design Facility menu:



Press ENTER once to send the updated screen layout to MANTIS. Then press the CANCEL key (be sure to press the CANCEL key only once) to exit the “Create or update a screen panel” and return to the Screen Design Facility menu. If you try to exit to the Facility Selection menu while you have any unsaved changes, MANTIS will display a warning message and ask for confirmation.

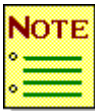
```

ASP001                                M A N T I S

                                     Screen Design Facility

Create or update a screen ..... 1
Update field specifications ..... 2
List field specifications ..... 3
Update repeat specifications ..... 4
List repeat specifications ..... 5
Display completed design ..... 6
Library functions ..... 7
Directory of screens ..... 8
Print completed design ..... 9
Terminate this facility ..... CANCEL

                                     :   :
  
```



You can move among the items listed on the Screen Design menu without losing your screen design, as long as you press ENTER before exiting from each option. You *must* use the screen design library functions to save your screen design before you exit to the MANTIS Facility Selection menu, or you will lose your work.

### Step 3: Using the screen design library functions

To exit from MANTIS at this point, you must first save your screen design. (If you want to proceed with the tutorial lesson, you do not have to save your design now.)

To save your design, enter 7 in the action field at the bottom of the Screen Design Facility menu and press ENTER. The Screen Design Library Facility screen displays:

ASP002

MANTIS

Screen Design Library Facility

Name of screen ... :

Password ..... :

Description ..... :

Map domain ..... : 22 80 : (Row,Column)

Blank fill character ... : | : Mask character ..... : # :

Sound alarm ..... : N : Full display ..... : N :

Protect bottom line .... : N : Opaque map ..... : N :

Save ..... 1

Replace ..... 2

Fetch ..... 3

Delete ..... 4

Terminate ..... CANCEL

:

:

The cursor is positioned at the beginning of the Name of screen field. Enter a symbolic name for the screen design: CUST\_BROWSE.

Use the TAB key to move the cursor to the Password field and enter a password for the screen.



For easy reference, enter your own user ID as the password for all the examples and exercises throughout this tutorial.

The password will not display as you type it. (A password is optional, and not required in order to save the screen design, or use it in a program.)



A screen name can be up to 30 characters in length; a password can be up to 16 characters.

Now, move the cursor to the beginning of the Description field. Supply a description of your screen: BURRYS CUSTOMER BROWSE SCREEN. This description helps you to identify the screen in your directory.



Keep names and descriptions short, accurate, and consistent for easy reference and use.

Let the other fields assume their default values:

Field name	Description	Default value
Map domain	Displays the row and column coordinates for the current screen domain	The size of your terminal screen
Blank fill character	Specifies the character you want to use to fill blank spaces	Vertical bar (   )
Mask character	Specifies the character you choose to use as the mask character to identify fields	#
Sound alarm	Indicates that you want to sound an alarm each time you converse the screen	No
Full display	Expands the screen size to the dimensions of the current terminal, including the bottom two lines of the screen	No
Protect bottom line	Protects the bottom line of the screen (the line that contains the command line and key simulation fields)	No
Opaque map	Indicates whether or not a screen (map) will be opaque (rather than transparent) when it is displayed	No

Your screen should look like this one:

```

ASP002                                M A N T I S

                                Screen Design Library Facility

Name of screen ... : cust_browse           :
Password ..... : password                 :
Description ..... : burrys customer browse screen :

Map domain ..... : 22 80 : (Row,Column)
Blank fill character ... : | :           Mask character ..... : # :
Sound alarm ..... : N :           Full display ..... : N :
Protect bottom line ... : N :           Opaque map ..... : N :

Save ..... 1
Replace ..... 2
Fetch ..... 3
Delete ..... 4
Terminate ..... CANCEL

: 1 :

```

Select the Save option by entering 1 in the action field. Press ENTER. MANTIS automatically returns you to the Screen Design Facility menu and displays a confirmation message in the bottom, left corner of the screen:

```

ASP001                                M A N T I S

                                Screen Design Facility

Create or update a screen ..... 1
Update field specifications ..... 2
List field specifications ..... 3
Update repeat specifications ..... 4
List repeat specifications ..... 5
Display completed design ..... 6
Library functions ..... 7
Directory of screens ..... 8
Print completed design ..... 9
Terminate this facility ..... CANCEL

:  :

FACSA2I: 'CUST_BROWSE' SAVED

```

To take a break in the lesson at this point, press the CANCEL key to exit to the MANTIS Facility Selection menu; then, press the CANCEL key again to exit MANTIS.

If you exit the Screen Design Facility at this point, you will need to fetch your screen design when you are ready to continue with the screen design lesson. To do so:

1. Select the Design a Screen option from the MANTIS Facility Selection menu.
2. Select the Library functions option.
3. Enter the name of your screen design (CUST\_BROWSE).
4. Select the Fetch option by entering 3 in the action field and then pressing ENTER). MANTIS returns you to the Screen Design Facility menu and displays a confirmation message in the bottom, left corner of the screen. The screen is now in your work area, and ready for modification.

Step 4: Updating field specifications

You must now provide specifications (characteristics) for the data fields you created. The cursor is positioned in the action field on the Screen Design Facility menu. To select Update field specifications, enter 2 in the action field and press ENTER. MANTIS displays your screen design:

B|U|R|R|Y|S

CUSTOMER|BROWSE

CUSTOMER| | | | |CUSTOMER| | | | |BRANCH| | | | |CREDIT| | | | |CREDIT

NUMBER| | | | |NAME| | | | |NUMBER| | | | |RATING| | | | |LIMIT

#####

#####

####

##

\$#####

#####

ASP040A: Enter field name; select by cursor position; enter 'HELP'

46

P39-5026-01

To define or alter field specifications, you can select a field in one of three ways:

- ◆ Place the cursor in a field and press ENTER.
- ◆ If you have already provided a name for the field, you can enter the field name on the command line and press ENTER.
- ◆ Use one of these PF key settings:

PF key	Function
PF1 or PF13	Selects the first field on the screen.
PF2 or PF14	Selects the next field (the field following the last processed field) on the screen.
PF3 or PF15	Presents undefined data fields for attribute update one at a time. (Cancel the function by pressing the CANCEL key.)
PF4 or PF16	Selects the last field on the screen.
PF5 or PF17	Selects the previous field (the field before the last field processed).
PF6 or PF18	<p>Lets you select fields from a table. MANTIS lists the fields in row/column order. Page through the list using the PF keys listed at the bottom of the table.</p> <p>Mark a field for update by placing an <i>S</i> in the selection column. You can also mark a range of fields by placing an <i>R</i> in the first and last fields in the range. (Ranges can cross multiple pages.) You can mark more than one individual field or field range for update.</p> <p>When you press ENTER, MANTIS presents each field in sequence for update.</p>
PF7 or PF19	Selects a range of fields to be processed one at a time. Position the cursor in the first field of the range and press PF7 or PF19; then, position the cursor in the last field of the range and press PF7 or PF19 again. MANTIS presents each field in sequence for update.
PF8 or PF20	Switches between Update Field Specifications (define attributes) and Repeat Field Specifications (specify horizontal and vertical repeats).



PF keys are particularly helpful when you are defining or altering attributes for large screen designs.

Because you are creating a new screen, all of your data fields are undefined (that is, they do not have symbolic names or attributes). Move the cursor to the customer number data field and press ENTER. When you select a field, MANTIS highlights the field and displays a window for you to define field specifications:

B|U|R|R|Y|S  
CUSTOMER|BROWSE

CUSTOMER| | | | |CUSTOMER| | | | |BRANCH| | | | |CREDIT| | | | |CREDIT  
NUMBER| | | | |NAME| | | | |NUMBER| | | | |RATING| | | | |LIMIT

#####

+-----+  
| Field Name : cust\_number : Length : 6 :  
| Row/Column : 7 3 :  
| Data Type : TXT : Intensity : NOR : Cursor : N :  
| Protected : N : Auto Skip : Y : Uppercase : Y :  
| Blinking : N : Reverse Video : N : Highlight : N :  
| Color : NO : Modified Tag : N : Detectable : N :  
| Box : L. N U. N O. N R. N : SO/SI : N :  
| Repeats : V. 14 1 H. : Extended Edit : N :  
+-----+

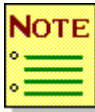
#####  
ASP030A: Modify the settings by overtyping and press 'ENTER'

The customer number field is highlighted and a boxed prompt lists the current (or default) attribute and repeat specifications for the field. MANTIS also displays the row/column coordinates and the length of the field.



You need to define the remaining attributes for this field. Make the following changes to the attribute specifications for the customer number field as shown in the preceding screen illustration. To do so, place the cursor in the field and enter the indicated data:

- ◆ **Field Name (CUST\_NUMBER).** Provides a name that identifies the data in this field. You can assign names to data fields only; not to heading fields. (For heading fields, the name is filled with a NOT REQUIRED message.) The symbolic name must follow the rules for MANTIS symbolic names (see “[Using symbolic names](#)” on page 115). The symbolic name must begin with an alphabetic character. It can contain 1-30 alphabetic and/or numeric characters, and the underline character *only*, and may not contain blank spaces or hyphens. (Note that the parts of the symbolic name CUST\_NUMBER are connected by an underline character so MANTIS interprets them as one name.)
- ◆ **Repeats (V. 14 1).** Entering 14 after the V indicates that you want 14 *additional* vertical occurrences of the field. The 1 indicates you want the repeats to be single-spaced on the screen. (To indicate double-spacing, you would enter 2 in this field.) MANTIS fits as many horizontal repeats as possible into a screen domain before wrapping to the next line. You can repeat numeric fields in both directions; however, text fields can only repeat in one direction. (You can also specify repeats using the Update repeat specifications option from the Screen Design Facility menu).



---

When specifying field repeats, it is important to stay within the boundaries of your screen design. If you specify more repeats than can fit on your screen design, the fields may overlay each other.

---

For the CUST\_NUMBER field, you will accept the default values for the remaining attributes. However, there are several other attribute fields that you should note for future reference:

- ◆ **Length.** Specifies the length of the field. You can alter the length of a text field on this screen to correct a length error in the design.
- ◆ **Row/Column.** Indicates the selected field's row/column position.
- ◆ **Intensity.** Specifies whether the field should be normal, bright, or hidden. The default value is NORMAL. If, however, you want a bright or hidden field, position the cursor over the N in NOR and enter B (BRIGHT) or H (HIDDEN). Hidden fields are often used for passwords; bright fields are used for headings or key fields.
- ◆ **Data Type.** Indicates whether the field will contain text, numeric, or Kanji data. MANTIS automatically supplies the data type for heading fields (HED). Fields with edit characters (that is, dollar sign, decimal point, etc.) are automatically numeric. For all other data fields, the default value is TEXT (TXT). Otherwise, you specify N (NUMERIC), or K (KANJI).
- ◆ **Protected.** Specifies whether you want this field protected (that is, read-only capability) or unprotected (that is, available for data entry). Heading fields are automatically protected. The default value is NO. If you want to protect a field, position the cursor over the N in NO and enter Y (YES).
- ◆ **Auto Skip.** Indicates that the cursor will skip automatically to the next unprotected data field when a user fills the current field. If autoskip is set to No, the cursor will remain on the field after the user fills it.

For a detailed description of the other available screen design attributes, refer to *MANTIS Facilities, OS/390, VSE/ESA*, P39-5001.

Accept the default values for the other attributes by pressing ENTER. MANTIS highlights the field you just defined.



---

You can request help information for this screen. To display a prompter that lists the PF key settings for this function, enter the HELP command on the command line and press ENTER. To view a prompter describing the attribute specifications, enter HELP(ATTRIBUTES) on the command line and press ENTER.

---

Now, use Pf2 to select the next data field. Supply a symbolic name for this field. (Call it CUST\_NAME.) Enter 14 and 1 for the vertical repeats and displacement values, respectively. Again, accept the defaults for the other attributes by pressing ENTER without modifying the default values:

```

      B|U|R|R|Y|S
      CUSTOMER|BROWSE

CUSTOMER| | | | |CUSTOMER| | | | |BRANCH| | | | |CREDIT| | | | |CREDIT
NUMBER| | | | |NAME| | | | |NUMBER| | | | |RATING| | | | |LIMIT

#####  #####  ####  ##  $#####

+-----+
| Field Name : cust_name                : Length : 20 :
|                                           Row/Column : 7  11 :
| Data Type : TXT :      Intensity : NOR :      Cursor : N :
| Protected : N :      Auto Skip : Y :      Uppercase : Y :
| Blinking : N :      Reverse Video : N :      Highlight : N :
| Color : NO :      Modified Tag : N :      Detectable : N :
| Box : L. N U. N O. N R. N :      SO/SI : N :
| Repeats : V. 14 1 H. :      Extended Edit : N :
+-----+

#####
ASP030A: Modify the settings by overtyping and press 'ENTER'

```

Use Pf3 to present the remaining undefined fields one at a time. Provide symbolic names and repeat specification for the next three undefined fields as shown in this table:

Heading	Symbolic name	Repeat specifications
BRANCH NUMBER	CUST_BRCH_NUMBER	V 14 1
CREDIT RATING	CUST_CREDIT_RAT	V 14 1
CREDIT LIMIT	CUST_CREDIT_LIMIT	V 14 1

Accept the default values for the other attributes by pressing ENTER after you supply the name and repeat specifications for each field.



The dollar sign (\$) in the CREDIT LIMIT field indicates to MANTIS that this field is masked and numeric (NUM).

Supply the name MESSAGE for the last data field on your screen. The message field appears only once on the screen, so there are no repeat specifications. Move the cursor to the Intensity field and enter a B (for BRIGHT) over the N in NOR. (Bright specifies that the MESSAGE data field will be displayed in high intensity.)

You should also specify that MESSAGE is a protected field, since you do not want the user to be able to alter the information that will appear there. Move the cursor to the Protected field and enter Y ( for YES) over the N:

B|U|R|R|Y|S  
CUSTOMER|BROWSE

CUSTOMER| | | | |CUSTOMER| | | | |BRANCH| | | | |CREDIT| | | | |CREDIT  
NUMBER| | | | |NAME| | | | |NUMBER| | | | |RATING| | | | |LIMIT

#####

Field Name : message : Length : 76 :  
Row/Column : 22 2 :  
Data Type : TXT : Intensity : bOR : Cursor : N :  
Protected : y : Auto Skip : Y : Uppercase : Y :  
Blinking : N : Reverse Video : N : Highlight : N :  
Color : NO : Modified Tag : N : Detectable : N :  
Box : L. N U. N O. N R. N : SO/SI : N :  
Repeats : V. H. : Extended Edit : N :

#####  
ASP030A: Modify the settings by overtyping and press 'ENTER'

Accept the default values for the other specifications by pressing ENTER without modifying any other values.

You have now defined all of the data fields for this screen, so press the CANCEL key to exit from this option and return to the Screen Design Facility menu.



Press ENTER once to send the *message* field specifications to MANTIS. Then press the CANCEL key (be sure to press the CANCEL key only once) to exit the “Update field specifications” panel and return to the Screen Design Facility menu. If you try to exit to the Facility Selection menu while you have any unsaved changes, MANTIS will display a warning message and ask for confirmation.

## Step 5: Listing field specifications

Now, to verify that you have defined all of the data fields as you intended, select the List field specifications option. (Enter 3 in the action field and press ENTER, or press PF3.) The following screen displays:

Attribute Listing	
CUST_NUMBER	( 7 , 3 ) , 6 , TXT , UPP , UNP , AUT
CUST_NAME	( 7 , 11 ) , 20 , TXT , UPP , UNP , AUT
CUST_BRCH_NUMBER	( 7 , 33 ) , 4 , TXT , UPP , UNP , AUT
CUST_CREDIT_RAT	( 7 , 48 ) , 2 , TXT , UPP , UNP , AUT
CUST_CREDIT_LIM	( 7 , 60 ) , 6 , NUM , MAS , UNP , AUT
MESSAGE	( 22 , 2 ) , 76 , TXT , UPP , PRO , BRI , AUT

MANTIS displays a list of the data field names in row/column order, along with the row/column coordinates, the field length, and the attribute specifications. For example, the CUST\_NUMBER field appears at row 7, column 3. It has a length of 6, is a TEXT field (TXT), displays in all uppercase letters (UPP), is unprotected (UNP), and has the AUTOSKIP attribute (AUT).



MANTIS supplies the MASKED attribute (abbreviated MAS above) for those data fields that contain characters other than #. Since CUST\_CREDIT\_LIM contains the dollar sign (\$), MANTIS assigns the MASKED attribute.

Your attributes should correspond with those in the preceding screen. If they do, press the CANCEL key to return to the Screen Design Facility menu.

If your attributes do not correspond with those in the preceding screen, press the CANCEL key to return to the Screen Design Facility menu, then select option 2 (Update field specifications) and correct the fields in error. (For detailed instructions on updating field specifications, see “[Step 4: Updating field specifications](#)” on page 46.)

## Step 6: Listing repeat specifications

Now, verify that you have defined your repeat specifications as you intended by selecting List repeat specifications. Enter 5 in the action field on the Screen Design Facility menu and press ENTER (or press PF5). MANTIS displays the repeat specification listing:

Repeat Specification			
ASP058I:CUST_NUMBER	occurs	15 displacement	1 vertical
ASP058I:CUST_NAME	occurs	15 displacement	1 vertical
ASP058I:CUST_BRCH_NUMBER	occurs	15 displacement	1 vertical
ASP058I:CUST_CREDIT_RAT	occurs	15 displacement	1 vertical
ASP058I:CUST_CREDIT_LIM	occurs	15 displacement	1 vertical

The MESSAGE field does not display here because it occurs only once on the screen, and so has no repeat specifications.

Your repeat specifications should correspond with those in the preceding screen. If they do, press the CANCEL key to return to the Screen Design Facility menu.

If your repeat specifications do not correspond with those in the preceding screen, press the CANCEL key to return to the Screen Design Facility menu, then select option 2 (Update field specifications) and correct the fields in error. (For detailed instructions on updating field specifications, see [“Step 4: Updating field specifications”](#) on page 46.)



If you have no changes to make to other attributes, you can also use option 4 (Update repeat specifications) to correct repeat specifications for a screen.

## Step 7: Displaying the completed screen design

You can display your final screen design before replacing it in your library by selecting the Display completed design option (number 6 or PF6) from the Screen Design Facility menu. When you do so, the following screen displays:

[illegible]

The screen shows a completed customer browse screen. Note that MANTIS does not display the row scale line or the blank fill characters (| ).

If the screen domain is larger than the physical screen, MANTIS automatically initiates window mode. In this case, you can use the window mode PF keys to scroll and view the entire design. (Window mode PF key settings and row/column coordinates display on the last line of the screen.)

Your completed screen design should correspond with the design in the preceding screen. Press the CANCEL key to return to the Screen Design Facility menu.



## Step 8: Saving the screen

You should now save or replace your completed screen in your library. To do so, select the Library functions option from the Screen Design Facility menu. If you have previously saved the screen, the name and description of the screen appear in the relevant fields:

```

ASP002                                M A N T I S

                                Screen Design Library Facility

Name of screen ... : CUST_BROWSE      :
Password ..... :      : password
Description ..... : BURRYS CUSTOMER BROWSE SCREEN      :

Map domain ..... : 22  80 : (Row,Column)
Blank fill character ... : | :      Mask character ..... : # :
Sound alarm ..... : N :      Full display ..... : N :
Protect bottom line .... : N :      Opaque map ..... : N :

                                Save ..... 1
                                Replace ..... 2
                                Fetch ..... 3
                                Delete ..... 4
                                Terminate ..... CANCEL

                                : 1 :

```

You can save a screen design only once. After that, you must use the Replace option to store any updates:

- ◆ **If you have *not* previously saved the screen:** Enter the symbolic name, a password (the password will not display as you type it, because it has a “hidden” attribute), and the description, as shown. Select the Save option by entering a 1 in the action field and pressing ENTER.
- ◆ **If you *have* previously saved the screen:** Move the cursor to the Password field and supply the password you assigned to the screen. (The password will not display as you type it.) Select the Replace option by entering a 2 in the action field and pressing ENTER.

MANTIS automatically returns you to the Screen Display Facility menu and displays a confirmation message in the bottom, left corner of the screen.

### Step 9: Viewing the directory of screens

Use the Directory of screens option to display an alphabetic listing of all existing screen designs. This list includes the first 16 characters of the screen name and the screen password, format (new or old), and description.

Select the Directory of screens option from the Screen Design Facility menu by entering 8 in the selection field and pressing ENTER, or by pressing PF8. MANTIS displays the directory of screens:

DIR003	Directory of Screens			YYYY/MM/DD
BURRYS				HH:MM:SS
-----Name-----	----Password----	Fmt	-----Description-----	
CUST_BROWSE	PASSWORD	NEW	BURRYS CUSTOMER BROWSE SCREEN	



MANTIS screens can have names up to 30 characters in length. To view the entire name, enter window mode by moving the cursor to the bottom, right corner of the directory list screen; entering w; and pressing ENTER. Press PF9 to exit window mode. (If you are printing the directory list, the extended list for screen names is automatically printed.)

Press ENTER to return to the Screen Design Facility menu.

## Step 10: Printing the completed screen design

If you would like to print a hard copy of your final screen design, select the Print completed design option from the Screen Design Facility menu. (Enter 9 in the action field and press ENTER, or press PF9.) The printout will be routed to your designated printer.



---

Your Master User designates your printer in your user profile. If you have no designated printer, MANTIS will display an error message if you try to print your screen.

---

You can return to the Screen Design Facility menu at any time during the screen design phase and select this option. MANTIS routes the current screen design to your designated printer.

When you select this option, a confirmation message displays to confirm that the screen has printed.

## Step 11: Using screen design window mode

In the previous sections of this chapter, we concentrated on the first of the four screens you will design for the Burrys scenario. You created a new screen, Burrys customer browse (CUST\_BROWSE), by following the order of functions on the Screen Design Facility menu.

Remember that a screen design is painted on a work area that contains 255 rows and 255 columns. You created the Burrys customer browse screen using a screen domain of 22 rows and 80 columns.

The Screen Design Facility provides its own window mode that allows you to design a screen that is larger than the size of your physical terminal. Screen design window mode is controlled by the following PF key settings:

PF key	Function
PF7 or PF19	Scrolls the screen up 10 lines
PF8 or PF20	Scrolls the screen down 10 lines
PF9 or PF21	Ends window mode
PF10 or PF22	Scrolls the screen left 20 columns
PF11 or PF23	Scrolls the screen right 20 columns
PF12 or PF24	Returns the screen to the origin (row 1, column 1)

Screen design window mode is always operative during the screen design process (option 1). You will now use window mode to add a Comments field to the CUST\_BROWSE screen. In the process, you will expand the screen, and practice using the screen commands and PF keys to modify and scroll around your screen.

To modify the screen, it must be in your screen design work area. If you have just re-entered the Screen Design Facility from the Facility Selection menu, you must fetch the screen from the library into your work area:

1. From the Screen Design Facility menu, select Library functions.
2. Enter the screen name, CUST\_BROWSE.
3. Select the Fetch option (3). MANTIS returns you to the Screen Design Facility menu and displays a confirmation message in the bottom, left corner of the screen.

Now, select the Create or update a screen option (1) from the Screen Design Facility menu. Your original design appears:

```

.                                     B|U|R|R|Y|S
.                                     CUSTOMER|BROWSE
.
.  CUSTOMER |||||CUSTOMER |||||BRANCH |||||CREDIT |||||CREDIT
+  NUMBER  |||||NAME  |||||NUMBER |||||RATING |||||LIMIT
.
.  #####  #####  #####  ##  $#####
.
.
1
.
.
.
.
+
.
.
.
.
2
.  #####
ASP026A: 'ENTER' to update, HELP for instructions, CANCEL to terminate
def

```

You are going to expand your screen to include a Comments field (length 25) next to the Credit Limit field. First, expand the size of the logical screen domain by changing the default values for your screen. (Remember that your screen domain defaults to the size of your terminal.)

Move the cursor to the command line and enter DEF (for DEFAULTS). Press ENTER. MANTIS displays a boxed prompt:

```

-----
Screen Design Row Domain ..... : 22 :
-----
Column Domain ..... : 80 :
-----
Blank Fill Character .... : | :
-----
Mask Character ..... : # :
Screen Paint Row Scrolling Increment .. : 10 :
-----
Column Scrolling Increment : 20 :
-----

```

ASP030A:Modify the settings by overtyping and press 'ENTER'

From this prompt, you can alter the default screen size (row and column domains), the default blank fill character, the default mask character, and the row and column scrolling increments.

Since you will add a field of 25 bytes, you must extend the screen. The Comments field needs a total of 25 columns. But, because 10 columns are already available on the screen, you only need to extend the screen by 15 columns.

Leave the Row Domain at 22 and change the Column Domain to 95, as shown:

```
+-----+
| Screen Design Row Domain ..... : 22 : |
| ----- Column Domain ..... : 95 : |
| ----- Blank Fill Character .... : | : |
| ----- Mask Character ..... : # : |
| Screen Paint Row Scrolling Increment .. : 10 : |
| ----- Column Scrolling Increment : 20 : |
+-----+
```

ASP030A:Modify the settings by overtyping and press 'ENTER'

Leave the values of the other fields as they are.



The row and column scrolling increments specify how much your screen will scroll when you use the PF keys in screen design window mode.

Press ENTER. MANTIS redisplay your screen design:

```

.                                     B|U|R|R|Y|S
.                                CUSTOMER|BROWSE
.
.  CUSTOMER| |||||CUSTOMER| ||||| |BRANCH| ||||| |CREDIT| ||||| |CREDIT
+   NUMBER| ||||| |||NAME| ||||| |NUMBER| ||||| |RATING| ||||| |LIMIT
.
.   #####  #####          #####          ##          $#####
.
.
.
1
.
.
.
+
.
.
.
.
2
. #####
ASP026A: 'ENTER' to update, HELP for instructions, CANCEL to terminate

```

Press PF11 to move your terminal window to the right:

```

.          B|U|R|R|Y|S
.          CUSTOMER|BROWSE
.
.  OMER| || || || |BRANCH| || || |CREDIT| || || ||CREDIT
+ ME| || || || || |NUMBER| || || |RATING| || || ||LIMIT
.
.  #####      ####          ##          $#####
.
.
1
.
.
.
.
+
.
.
.
.
.
2
.  #####
ASP026A: 'ENTER' to update, HELP for instructions, CANCEL to terminate

```





Remember, MANTIS reserves the first three columns on the screen for the row scale line. The last two lines are reserved for the message line, row/column coordinates, command line, and the key simulation field. These fields remain on your screen even when you move your window.

With your cursor positioned anywhere on the same line as "BURRYS", press PF3 to display the column scale line. It will overlay the top line of your screen, making it temporarily invisible:

```

.....+.....3.....+.....4.....+.....5.....+.....6.....+.....7.....+.....8.....+.....9.....+
      CUSTOMER|BROWSE
.
.  OMER| || || || |BRANCH| || || || |CREDIT| || || || |CREDIT
+ ME| || || || || |NUMBER| || || || |RATING| || || || |LIMIT
.
. #####      ####          ##          $#####
.
.
1
.
.
.
.
+
.
.
.
.
2
. #####
ASP026A: 'ENTER' to update, HELP for instructions, CANCEL to terminate

```

Later, you will remove the column scale line, and the top line of your screen will display again.

The current column position within the work area is indicated by the column scale line. This scale line is adjusted as your terminal window moves across the work area.

With your cursor next to the word “LIMIT” (in the CREDIT LIMIT heading), enter vertical bars up to position 70. With your cursor at column 70, add the heading and the 25 hash signs for the data characters of the Comments field:

```
. ....+....3....+....4....+....5....+....6....+....7....+....8....+....9....+
.          CUSTOMER|BROWSE
.
.  OMER ||||| |BRANCH| ||||| |CREDIT| ||||| |CREDIT
+ ME ||||| |NUMBER| ||||| |RATING| ||||| |LIMIT| ||| |-----COMMENTS-----|
.
. #####      ####          ##          $#####  #####
.
.
1
.
.
.
.
+
.
.
.
2
. #####
ASP026A: 'ENTER' to update, HELP for instructions, CANCEL to terminate
```

Press ENTER to save the screen modifications in your work area.

When you are finished entering the Comments field, position the cursor anywhere on the column scale line and press PF3. The column scale line is removed from your screen, and the top line of your screen is visible again:

```

.          B|U|R|R|Y|S
.      CUSTOMER|BROWSE
.
.  OMER ||||| |BRANCH| ||||| |CREDIT| ||||| |CREDIT
+ ME ||||| |NUMBER| ||||| |RATING| ||||| |LIMIT| |||-----COMMENTS-----
.
. #####      ####          ##          $#####      #####
.
.
1
.
.
.
.
+
.
.
.
.
2
. #####
ASP026A: 'ENTER' to update, HELP for instructions, CANCEL to terminate

```

Press PF10 to scroll your terminal window back to the left:

```
.
.                                     B|U|R|R|Y|S
.                                CUSTOMER|BROWSE
.
.   CUSTOMER| |||||CUSTOMER| |||||BRANCH| |||||CREDIT| |||||CREDIT
+    NUMBER| |||||NAME| |||||NUMBER| |||||RATING| |||||LIMIT| |||-----C
.
.   #####      #####          ##              $#####      #####
.
.
. 1
.
.
.
.
+
.
.
.
.
2
. #####
ASP026A: 'ENTER' to update, HELP for instructions, CANCEL to terminate
```



You could also have pressed PF12 to return the screen to its origin (row 1, column 1).

Now, you need to specify the field attributes for the Comments field that you just added to your screen design. Press the CANCEL key to return to the Screen Design Facility menu, then select the Update field specifications option.

MANTIS displays your screen design. Press PF3 or PF15 to display the undefined fields (in this case, the Comments field). The Comments field is highlighted and a boxed prompt lists the current specifications for that field:

```

.      B|U|R|R|Y|S
.      CUSTOMER|BROWSE

OMER||| ||| |BRANCH||| ||| |CREDIT||| ||| |CREDIT
+ ME||| ||| |NUMBER||| ||| |RATING||| ||| |LIMIT||| |-----COMMENTS-----+

. #####      ####      ##      $#####      #####
.
.
.

+-----+
| Field Name :                               : Length : 25 :
|                                         Row/Column : 7   70 :
| Data Type : TXT :      Intensity : NOR :      Cursor : N :
| Protected : N :      Auto Skip : Y :      Uppercase : Y :
| Blinking : N :      Reverse Video : N :      Highlight : N :
| Color : NO :      Modified Tag : N :      Detectable : N :
| Box : L. N U. N O. N R. N :      SO/SI : N :
| Repeats : V.      H.      :      Extended Edit : N :
+-----+

. #####
ASP030A: Modify the settings by overtyping and press 'ENTER'

```



If a selected field is outside the boundary of the current window display, MANTIS moves the window so you can view the field that you are updating.

Enter CUST\_COMMENTS for the field name and specify the necessary vertical repeats. To do so, move the cursor to Repeats. After the V (vertical), enter 14 for the number of vertical repeats, and 1 for the vertical displacement:

```

.      B|U|R|R|Y|S
.      CUSTOMER|BROWSE

.      OMER| ||||| |BRANCH| ||||| |CREDIT| ||||| |CREDIT
+ ME| ||||| |NUMBER| ||||| |RATING| ||||| |LIMIT| ||| -----COMMENTS-----
.
. #####      ###      ##      $#####      #####
.
.
+-----+
| Field Name : cust_comments                               : Length : 25 :
|                                         Row/Column : 7   70 :
| Data Type : TXT :      Intensity : NOR :      Cursor : N :
| Protected : N :      Auto Skip : Y :      Uppercase : Y :
| Blinking : N :      Reverse Video : N :      Highlight : N :
| Color : NO :      Modified Tag : N :      Detectable : N :
| Box : L. N U. N O. N R. N :      SO/SI : N :
| Repeats : V. 14 1 H. :      Extended Edit : N :
+-----+

.#####
ASP030A: Modify the settings by overtyping and press 'ENTER'

```

Press ENTER.



Remember, MANTIS supplies default values for all the other attributes if you do not specify any new values.

Press the CANCEL key to return to the Screen Design Facility menu.

Save your modifications by selecting the Library functions option. The screen name and description are provided automatically by MANTIS as shown:

```

ASP002                                M A N T I S

                                Screen Design Library Facility

Name of screen ... : CUST_BROWSE
Password ..... :
Description ..... : BURRYS CUSTOMER BROWSE SCREEN

Map domain ..... : 22 95 : (Row,Column)
Blank fill character ... : | : Mask character ..... : # :
Sound alarm ..... : N : Full display ..... : N :
Protect bottom line .... : N : Opaque map ..... : N :

                                Save ..... 1
                                Replace ..... 2
                                Fetch ..... 3
                                Delete ..... 4
                                Terminate ..... CANCEL

                                :
                                :
```

Enter your password, and select the Replace option (enter 2 and press ENTER, or press PF2).



Notice that the value for the column domain is now 95, instead of 80.

MANTIS returns you to the Screen Design Facility menu and displays a confirmation message in the bottom, left corner of the screen:

```
ASP001                      M A N T I S

                             Screen Design Facility

Create or update a screen ..... 1
Update field specifications ..... 2
List field specifications ..... 3
Update repeat specifications ..... 4
List repeat specifications ..... 5
Display completed design ..... 6
Library functions ..... 7
Directory of screens ..... 8
Print completed design ..... 9
Terminate this facility ..... CANCEL

                             :   :

FACSA2I: 'CUST_BROWSE' REPLACED
```

To display your updated screen, select the Display completed design option from the Screen Design Facility menu.



MANTIS automatically places you in screen display window mode because you increased the screen domain beyond the boundaries of the physical screen. Notice that the prompt for PF keys appears at the bottom of your screen:

[illegible]

You can use the PF keys to scroll around the screen display.

Because you are in the display option, the row/column coordinates appear at the bottom, right corner of the screen. They specify the upper left corner of the physical screen's position on the work area.



You can also scroll around the screen by changing the values of the row/column coordinates. To do so, enter the new position for the top left corner of the logical screen and press ENTER.

Press PF11 to scroll your terminal screen to the right. The entire  
Comments field is now visible:

B U R R Y'S				
CUSTOMER BROWSE				
USTOMER NAME	BRANCH NUMBER	CREDIT RATING	CREDIT LIMIT	-----COMMENTS-----
#####	####	##	\$#####	#####
#####	####	##	\$#####	#####
#####	####	##	\$#####	#####
#####	####	##	\$#####	#####
#####	####	##	\$#####	#####
#####	####	##	\$#####	#####
#####	####	##	\$#####	#####
#####	####	##	\$#####	#####
#####	####	##	\$#####	#####
#####	####	##	\$#####	#####
#####	####	##	\$#####	#####
#####	####	##	\$#####	#####
#####	####	##	\$#####	#####
#####	####	##	\$#####	#####
#####				
NUCWMMA: PF7-UP PF8-DOWN PF9-END PF10-LEFT PF11-RIGHT PF12-HOME 001 021				

To return the screen to the origin, press PF12:

[illegible]

Notice that the row/column coordinates regained their original values: 001 001.

Now, press PF6 or CANCEL to return to the Screen Design Facilities menu and proceed with the exercises in the next section.

## Exercises

Three additional screens are needed for the Burrys application:

- ◆ Burrys customer accounts menu (CUST\_MENU)
- ◆ Burrys new customer entry screen (CUST\_ENTRY)
- ◆ State code entry screen (STATE\_CODE)

Create these screens using the procedures outlined in this chapter and the data supplied with each exercise.

To begin:

1. Press the CANCEL key to return to the Facility Selection menu from the Screen Design Facility menu. This clears the Burrys customer browse screen from your work area. (Remember, your design has been permanently saved in your library.)
2. Select the Design a Screen option. When the Screen Design Facility menu appears, you are ready to begin the first exercise.

Our versions of the completed screens appear in “[Exercise examples](#)” on page 237.

## Exercise 1: Creating the customer menu screen

Create the customer accounts menu to correspond with the following screen design:

B		U		R		R		Y		S	
CUSTOMER				ACCOUNTS				SYSTEM			
ENTER		A		NEW		CUSTOMER				.....  1	
BROWSE		CUSTOMER		LIST						.....  2	
EXIT		THIS		FACILITY						.....  CANCEL	
:		#		:							

Notice that a space appears before and after the hash sign (#). At least one space must separate fields (except when you use the blank fill character as a nondisplaying character for heading fields).



Remember to press ENTER to save your screen design temporarily.

Press the CANCEL key to return to the Screen Design Facility menu; then, proceed with Update field specifications. (For step-by-step directions for adding field specifications, see [“Step 4: Updating field specifications”](#) on page 46.)

The symbolic name (supplied next to Field Name) for the one-byte date field (between the colons) should be ACTION. The field has a data type of NUMERIC:

```
+-----+
| Field Name : action                               : Length : 1 :
|                                         Row/Column : 18 37 :
| Data Type : nXT :      Intensity : NOR :      Cursor : N :
| Protected : N :      Auto Skip : Y :      Uppercase : Y :
| Blinking  : N :      Reverse Video : N :      Highlight : N :
| Color     : NO :      Modified Tag : N :      Detectable : N :
| Box       : L. N U. N O. N R. N :      SO/SI : N :
| Repeats   : V.      H.      :      Extended Edit : N :
+-----+
VIEW|CUSTOMER|BROWSES|SCREEN .....|2
EXIT|THIS|FACILITY .....|CANCEL

: # :
```

Press ENTER to update the field specifications.

Press the CANCEL key, and MANTIS returns you to the Screen Design Facility menu. Proceed with the sequence of functions listed on the menu.



This screen needs no repeat specifications.

Save your design (via library functions), specifying a name of CUST\_MENU, a password, and the description “BURRYS CUSTOMER ACCOUNTS MENU”.

## Exercise 2: Creating the new customer entry screen

Create the new customer entry screen to correspond with the following screen design:

```

      B|U|R|R|Y|S
    NEW|CUSTOMER|ENTRY

NAME: #####
ADDRESS: #####
CITY: #####
STATE: ##
ZIP|CODE: #####

CUSTOMER|NUMBER: #####
CLASS: ##
CUSTOMER|CREDIT|RATING: ##
CREDIT|LIMIT: #####
BRANCH|NUMBER: #####
COMMENTS: #####

#####
  
```

Finish the screen design process using the following data specifications:

Heading	Symbolic name	Field length	Attributes
NAME	CUST_NAME	20	Autoskip
ADDRESS	CUST_ADDRESS	20	Autoskip
CITY	CUST_CITY	13	Autoskip
STATE	CUST_STATE	2	Autoskip
ZIP CODE	CUST_ZIP_CODE	5	Numeric, autoskip
CUSTOMER NUMBER	CUST_NUMBER	6	Autoskip
CLASS	CUST_CLASS	2	Autoskip
CUSTOMER CREDIT RATING	CUST_CREDIT_RAT	2	Autoskip
CREDIT LIMIT	CUST_CREDIT_LIM	5	Numeric, autoskip
BRANCH NUMBER	CUST_BRCH_NUMBER	4	Autoskip
COMMENTS	CUST_COMMENTS	25	Autoskip
	MESSAGE	76	Bright, protected

When you are finished creating your screen, save your design using the library functions. Specify a name of CUST\_ENTRY, a password, and the description “BURRYS NEW CUSTOMER ENTRY SCREEN”.

**Exercise 3. Creating the state code entry screen**

Create the state code entry screen to correspond with the following screen design:

STATE | CODE | ENTRY

: ## :

Proceed with the rest of the screen design process as presented on the Screen Design Facility menu. The symbolic name for the two-byte data field should be CUST\_STATE.

Save your design via library functions. Specify the name STATE\_CODE, enter a password, and supply the description “STATE CODE ENTRY SCREEN”.



## Exercise 4: Viewing the directory of screens

When you have finished designing and saving these screens, select the Directory of screens option from the Screen Design Facility menu. (Enter 8 and press ENTER, or press PF8.) Your Directory of Screens should correspond with the following screen:

Directory of Screens			YYYY/MM/DD
			HH:MM:SS
-----Name-----	----Password----	Fmt	-----Description-----
CUST_BROWSE	PASSWORD	NEW	BURRYS CUSTOMER BROWSE SCREEN
CUST_ENTRY	PASSWORD	NEW	BURRYS NEW CUSTOMER ENTRY SCREEN
CUST_MENU	PASSWORD	NEW	BURRYS CUSTOMER ACCOUNTS MENU
STATE_CODE	PASSWORD	NEW	STATE CODE ENTRY SCREEN

Press the CANCEL key to return to the Screen Design Facility menu.



If there is more than one page of screens, press ENTER to get the next page. After MANTIS lists the last screen in the directory and you press ENTER, MANTIS will return to the Screen Design Facility menu.

You can enter a partial screen name in the Command Line (lower left-hand corner of the screen). The directory list repositions itself to the screen name matching the partial screen name you enter. If the entered partial screen name matches no screen name in the directory list, the directory list repositions itself to the next valid screen name alphanumerically following the entered partial screen name.

You have now completed the screen design portion of the Burrys scenario. To continue with the tutorial, proceed to “[Creating a MANTIS file](#)” on page 83.



# 3

---

## Creating a MANTIS file

---

In this chapter you will create the Burrys customer information file, which is one of two files used in the Burrys scenario. When you create MANTIS programs later in this tutorial, the records in this file will be read and displayed on the Burrys customer browse screen (CUST\_BROWSE) that you designed in chapter 2.

---

### Learning outline

In this chapter, you will learn how to perform the following:

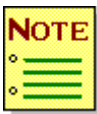
- ◆ Specify a name and define the access levels for a file.
- ◆ Create or update the record layout for a file, defining the fields it contains and the format in which MANTIS stores the data.
- ◆ Save a new file, or changes to an existing file, using the File Design Library Facility.
- ◆ View the directory of files.
- ◆ Print the completed file design.

---

## Basic concepts: Understanding MANTIS file design

Use the MANTIS File Design Facility to create, save, update, and maintain file views for internal MANTIS files. A *file view* contains detailed information about the contents and format of the data stored in a file, and allows you to control access to a file by password-protecting certain functions performed on the file data.

Use the Update record layout option to create or update the record layout for a file. A *record layout* defines the format in which MANTIS stores and transmits file data. The record layout for a file can contain up to 64 fields. (Each page of the record layout displays 16 fields.)



---

In our application, the element names in the MANTIS record layout definition must be exactly the same as the symbolic names of the corresponding data fields in your screen design.

---

Each element receives a data type of TEXT, BIG, SMALL, or KANJI. For TEXT and KANJI items, the length must be specified in the Dimensions field. You do not specify a length for SMALL and BIG elements.

Elements can have these attributes:

- ◆ **KEY.** Records in the file are ordered and accessed by the key element. Each record must have one or more key elements, and they must be the first elements in the record. The combined key lengths can be up to 32 bytes.
- ◆ **SCRAMBLE.** Indicates that the data in the field will be stored in an encrypted format in the file, for security purposes.

## Step-by-step: Creating the customer information file

Now that you understand some basic concepts for MANTIS file design, you're ready to design your first MANTIS file.

To begin, select the Design a File option from the MANTIS Facility Selection menu. The File Design Facility menu displays:

```
MFV001                                M A N T I S

                                     File Design Facility

Create or update file profiles ..... 1
Update record layout ..... 2
Library functions ..... 3
Directory of file profiles ..... 4
Print completed design ..... 5
Terminate this facility ..... CANCEL

                                     :   :
```

The File Design Facility menu presents the options in the order you perform them when you create a new MANTIS file view. If you are updating a file view, you must first select the Library functions option to fetch the file view from your library, then update the information as desired.

You can move among the File Design Facility menu options without losing the file view design currently in your work area.



You must use the library functions to save your new or updated file before exiting from the File Design Facility. If you attempt to exit from the facility without saving your changes, MANTIS asks you to confirm your exit. If you exit without saving, you lose your file design (if it is new), or any changes you have made since you last saved or replaced the design.

Step 1: Creating the file

Since you are designing a new file, you will follow the sequence of functions as they are listed on the File Design Facility menu (that is, first create a file profile, then update the record layout, etc.). In this step, you will specify the general characteristics for the file.

Select the Create or update file profiles option by entering 1 in the action field and pressing ENTER. The File Design Facility screen displays, with the cursor in the Name and description of file field. Enter the file information as it appears on the following screen:

MFV002

MANTIS

File Design Facility

Name and description of file : cust\_info :

: burrys customer information file :

Associated record layout :

Password for viewing :

Password for altering :

Password for deleting/inserting : password :

Status : active :

Internal file code : Dec: Hex:

Last profile update date : YYYY/MM/DD :

Last profile update time : :

Element count : :

Byte count : :

A file name can contain 1-16 alphanumeric characters, and a file description can contain 1-32 alphanumeric characters. The description helps you to identify the file in your file directory.

Do not supply an entry for Associated record layout. (You would supply an associated record layout name if you were using an identical record layout from another file, but in this case you will be creating a new record layout.)

It is not necessary to specify a password for each of the three password levels unless different passwords are desired for security reasons. (The Deleting/inserting specification includes privileges for altering and viewing, and the Altering specification includes privileges for viewing.)

Enter ACTIVE in the Status field. Entering anything other than ACTIVE (for example, ACTIVEb or OBSOLETE) prevents a program from accessing the file.

MANTIS maintains the last five fields (Internal file code through Byte count) on the File Design Facility screen.

When you have completed the entries as shown in the previous screen, press ENTER. MANTIS accepts your entries and automatically returns you to the File Design Facility menu:

```
MFV001                                M A N T I S

                                     File Design Facility

Create or update file profiles ..... 1
Update record layout ..... 2
Library functions ..... 3
Directory of file profiles ..... 4
Print completed design ..... 5
Terminate this facility ..... CANCEL

                                     :   :
```

Step 2: Updating the record layout

Now you must create a record layout for the file, to specify the format in which MANTIS stores and transmits file data. The record layout for a file can contain up to 64 fields, and each page of the record layout displays 16 fields.

Select the Update record layout option from the File Design Facility menu by entering 2 in the action field and pressing ENTER. The MANTIS Record Layout Definition screen displays, with the cursor positioned in the Page field:

MFV003

MANTIS Record Layout Definition

YYYY/MM/DD

Name: CUST\_INFO

HH:MM:SS

Page 1

Element Count

Size 32

Element -----Name----- Data-type Dimensions ----Attributes----

- - - - -

(Use PF1 - PF4 to page; use CANCEL to exit)

MANTIS supplies the Name (CUST\_INFO) you specified on the File Design Facility screen, along with the Date, Time, Page, Element Count, and Size fields.

The underline characters in the preceding screen represent the built-in tabs for this option. Use the TAB key to move from field to field and enter the specified data at the tab positions.

When entering or updating an element in the record layout, you must begin each line with an A, I, or D to indicate the action you want to perform (for Alter, Insert, or Delete respectively).



Begin by entering the specifications for the first element as they appear in the following screen:

```
MFV003                      MANTIS Record Layout Definition                      YYYY/MM/DD
Name: CUST_INFO                      HH:MM:SS
Page 1                      Element Count                      Size 32
Element  -----Name-----  Data-type  Dimensions  ----Attributes----
i   1   cust_number          t           6           k
```

(Use PF1 - PF4 to page; use CANCEL to exit)

To enter the specifications for the first element:

1. Use the TAB key to move the cursor from the Page field to the first tab position on line 1.
2. Enter an i to indicate that you are inserting a new line. The cursor automatically moves to the next tab position.
3. Enter 1 to indicate the line number; then, press TAB to move to the Name field.
4. Enter cust\_number in the Name field and press TAB again to move to the Data-type field.

5. You can specify TEXT, BIG, SMALL, or KANJI (T, B, S, or K) for the data type of each element. When selecting the data type, use:
  - ◆ TEXT for all alphanumeric fields (requires you to enter a length for the field under Dimensions)
  - ◆ BIG for a numeric field of up to 14 significant digits (recommended when using decimals)
  - ◆ SMALL for a numeric field of up to 6 significant digits (normally, an integer field)
  - ◆ KANJI for a Kanji data field (for Asian language support; requires you to enter a length for the field under Dimensions)

Since CUST\_NUMBER is a text field, enter T in the Data-type field; then, press TAB to move to the Dimensions field.

6. Enter 6 in the Dimensions field; then, press TAB twice to move to the Attribute field. (Use the second tab in the Dimensions field only when you are defining a list of fields, or an array.)
7. Enter K in the Attribute field to indicate that CUST\_NUMBER is the key to this file (that is, records in this file are ordered and accessed by CUST\_NUMBER). The key field(s) must be defined first in the list of fields.

Enter the remaining elements of the record layout as they appear in the following screen:

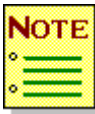
```

MFV003                      MANTIS Record Layout Definition                      YYYY/MM/DD
Name: CUST_INFO                      HH:MM:SS
Page 1                      Element Count                      Size 32
Element  -----Name-----  Data-type  Dimensions  ----Attributes----

i   1   cust_number          t           6           k
i   2   cust_name            t           20
i   3   cust_address         t           20
i   4   cust_city            t           13
i   5   cust_state           t           2
i   6   cust_zip_code        s
i   7   cust_class           t           2
i   8   cust_credit_rat      t           2
i   9   cust_credit_lim      s           S
i  10   cust_comments        t          25
i  11   cust_brch_number     t           4

(Use PF1 - PF4 to page; use CANCEL to exit)

```



The element names that you supply here must be identical to the symbolic names you gave to the corresponding data fields during screen design. Using identical symbolic names will enable data to be passed directly from this file to data fields on the customer browse screen without extra programming.

Notice that you supply dimensions for the text fields only. (Using a data type of BIG or SMALL automatically specifies the length of numeric fields.)

For CUST\_CREDIT\_LIM, supply the attribute S (SCRAMBLE) to indicate that the data in this field will be stored in an encrypted format.

After you have entered all of the data, press ENTER. MANTIS temporarily stores the data in your work area and updates the element count and size:

MFV003	MANTIS Record Layout Definition			YYYY/MM/DD
Name: CUST_INFO				HH:MM:SS
Page 1	Element Count 11		Size 138	
Element	-----Name-----	Data-type	Dimensions	----Attributes----
1	CUST_NUMBER	TEXT	6	KEY
2	CUST_NAME	TEXT	20	
3	CUST_ADDRESS	TEXT	20	
4	CUST_CITY	TEXT	13	
5	CUST_STATE	TEXT	2	
6	CUST_ZIP_CODE	SMALL		
7	CUST_CLASS	TEXT	2	
8	CUST_CREDIT_RAT	TEXT	2	
9	CUST_CREDIT_LIM	SMALL		SCRAMBLE
10	CUST_COMMENTS	TEXT	25	
11	CUST_BRCH_NUMBER	TEXT	4	
(Use PF1 - PF4 to page; use CANCEL to exit)				

Press the CANCEL key to return to the File Design Facility menu.

### Step 3: Saving the file

To save the file, select Library functions by entering 3 in the action field and pressing ENTER. The File Design Library Facility displays:

```

M A N T I S

File Design Library Facility

Name of file ..... : CUST_INFO      :
Internal file code ..... Dec:      Hex:

Save ..... 1
Replace ..... 2
Fetch ..... 3
Delete ..... 4
Terminate ..... CANCEL

: 1 :
```

MANTIS displays the name of the file in the Name field, so you do not need to enter it. To save the file, select the Save option (enter 1 and press ENTER, or press PF1).

MANTIS automatically returns you to the File Design Facility menu and displays a confirmation message in the lower, left corner of the screen.

Step 4: Viewing the directory of files

Use the Directory of file profiles option to display an alphabetic listing of all existing files. To display the list, select this option from the File Design Facility menu by entering 4 in the selection field and pressing ENTER (or by pressing PF4). MANTIS displays the Directory of MANTIS Files screen:

DIR002	Directory of MANTIS Files		YYYY/MM/DD
BURRYS			HH:MM:SS
-----Name-----	-----Status-----	-----Description-----	
CUST_INFO	ACTIVE	BURRYS CUSTOMER INFORMATION FILE	



If your directory has entries filling more than one page, you can page through them by pressing ENTER.

In addition to the CUST\_INFO file that you just created, you will see other files listed in the directory. Some of these have a description of “Cincom use only”. These files are used by MANTIS to save the entities that you design.

Press ENTER to return to the File Design Facility menu.

Step 5: Printing the completed file design

If you would like to view a printed copy of the file design, select the Print completed design option by entering 5 in the action field and pressing ENTER. The printout will be routed to your designated printer.

## Exercise

In this chapter, you created a new file, the Burrys customer information file, by following the order of functions presented on the File Design Facility menu. One other file is needed for the Burrys scenario: the state codes file.

Use CANCEL to exit to the Facility Selection menu in order to clear your work area. Again, select the Design a File option. Create the state codes file by following the same procedures that you used to create the previous file, but supply the following data on the File Design Facility screen:

Field	Data
File name	STATE_CODES
Description	State codes file
Password for deleting/inserting	PASSWORD
Status	ACTIVE

The record layout contains one element: CUST\_STATE. It is a text element with a dimension of 2. It is also the key element. The final file design should look like this:

```

MFV003                      MANTIS Record Layout Definition                      YYYY/MM/DD
Name: STATE_CODES                      HH:MM:SS
Page 1                      Element Count 1                      Size 32
Element  -----Name-----  Data-type  Dimensions  ----Attributes----
      1    CUST_STATE      TEXT      2      KEY

(Use PF1 - PF4 to page; use CANCEL to exit)

```

After you design and save the file, check the Directory of MANTIS Files:

```

Directory of MANTIS Files
                                YYYY/MM/DD
BURRYS                                HH:MM:SS
-----Name-----  -----Status-----  -----Description-----
CUST_INFO           ACTIVE                BURRYS CUSTOMER INFORMATION FILE
STATE_CODES         ACTIVE                STATE CODES FILE

```

Press the CANCEL key to return to the File Design Facility menu. Press the CANCEL key again to exit to the MANTIS Facility Selection menu. Then, proceed with “**Creating a prompter**” on page 97.



# 4

## Creating a prompter

In this chapter, you will design the state codes prompter for the Burrys scenario, which displays the two-letter abbreviation for each state. You will use this prompter with the customer entry program that you will create later in the tutorial.

### Learning outline

In this chapter, you will learn how to perform the following:

- ◆ Accept default tab values, or set new tabs for prompter design.
- ◆ Change the prompter design default tab character.
- ◆ Enter, update, and delete prompter text.
- ◆ Save a new prompter, or changes to an existing prompter, using the Prompter Design Library Facility.
- ◆ View the directory of prompters.
- ◆ View the completed prompter design.
- ◆ Print the completed prompter design.

## Basic concepts: Understanding prompter design

You can use prompters to present online help information to users. This includes online help for programs, files, and screens. You can also use prompters to document certain aspects of company policy or procedure.

The data on prompter screens is static, and remains the same except for occasional updates.

A MANTIS prompter allows up to 80 lines of text. Each screen displays 20 lines of the prompter, so you can have up to four pages of text per prompter. If this is not enough space, you can add pages by chaining to another prompter (specified in the Chain to next prompter field in library functions).

If your terminal provides lowercase support for text characters, MANTIS accepts the text you enter in a prompter as you enter it (any combination of uppercase and lowercase characters). This capability also depends upon CICS and MANTIS settings to work. Check with your Master User to find out if this capability is available in your environment.

If your terminal supports lowercase, you can enter the description and password for a prompter using lowercase characters at the Prompter Design Facility menu. The description entered here becomes the title of the prompter.

# Step-by-step: Creating the state codes prompter

Now that you understand some basic concepts of MANTIS prompter design, you're ready to design your first MANTIS prompter.

To begin, select the Design a Prompter option from the MANTIS Facility Selection menu. The Prompter Design Facility menu displays:

```
PRD001                                M A N T I S

                                     Prompter Design Facility

Create or update a prompter ..... 1
Set tabs ..... 2
Library functions ..... 3
Directory of prompters ..... 4
Display completed design ..... 5
Print completed design ..... 6
Terminate this facility ..... CANCEL

                                :   :
```

The Prompter Design Facility menu presents the options in the order in which you perform them when you create a new MANTIS prompter. (If you are updating an existing prompter, you must first select the Library functions option to fetch the prompter from your library, then update the information as desired.)

You can move among the Prompter Design Facility menu options without losing the prompter design currently in your work area.



You must use the library functions to save the new or updated prompter before you exit from the Prompter Design Facility. If you attempt to exit from the facility without saving current changes, MANTIS asks you to confirm your exit. If you exit without saving, you lose your prompter (if it is new), or any changes you have made since you last saved the prompter.

### Step 1: Setting the tabs for prompter design

When you are designing a new prompter, you can follow the sequence of functions as they appear on the Prompter Design Facility menu, with one exception: if you want to change the default tab character and positions, you must use the Set tabs option first.

Tabs make it easy for you to format a prompter. You simply enter each line of text, indicating a tab character when you want MANTIS to skip to the next tab position. When you press ENTER, MANTIS moves the text to the indicated position(s). (You will see how the tab settings work as you create the state codes prompter.)

The scale line across the top of the prompter design work area reflects the tab settings specified for the prompter design currently in the work area. This screen illustrates the default tab settings for prompter design:

PRD003

Page = 1

MANTIS Prompter Design Facility

Lines =

....

|.@..1....

|.@..2....

|.@..3....

|.@..4....

|.@..5....

|.@..6....

|.@..7....

|..

The at sign (@) is the default tab character, and it displays in the default tab positions at columns 7, 17, 27, 37, 47, 57 and 67.

The default tab positions will not work for the state codes prompter, since it requires a three-column format. Therefore, you must begin by selecting the Set tabs option from the Prompter Design Facility menu.

You can use the Set tabs option to:

- ◆ Change the default tab character.
- ◆ Change the default tab positions to reflect the format you need for your prompter design.

Select the Set tabs option from the Prompter Design Facility menu by entering 2 in the action field and pressing ENTER. MANTIS displays the following screen:

```
PRD004                                M A N T I S

                                     Prompter Design Facility

Tab character ..... @
Tab positions ..... 7
                                     17
                                     27
                                     37
                                     47
                                     57
                                     67
```

Notice that the default positions are listed on this screen, and that they correspond with those shown in the prompter design work area.

For the state codes prompter, you must change both the tab character (from @ to /) and settings (shown below) by entering the new tab character and settings over the old ones as indicated:

PRD004

M A N T I S

Prompter Design Facility

Tab character

.....

/

Tab positions

.....

29

57



Use the space bar or Erase EOF key to erase the extra five entries.

When you have made the changes indicated above, press ENTER.  
MANTIS automatically returns you to the Prompter Design Facility menu.

## Step 2: Creating the prompter

Now that the tabs are specified correctly, you are ready to select the Create or update a prompter option. This option lets you create a new prompter design or update an existing prompter design.

To select the Create or update a prompter option from the Prompter Design Facility menu, enter 1 in the action field and press ENTER. MANTIS displays the prompter design work area:

```
PRD003  Page = 1                MANTIS Prompter Design Facility      Lines =  
....|....1....|....2....|.../3....|....4....|....5....|./..6....|....7....|..
```

Notice that the scale line across the top of the prompter design work area now reflects the new tab character (/) and tab settings (29 and 57).

To manipulate information in the prompter design work area, you use the A (Alter), I (Insert), and D (Delete) action indicators, as you did during file design.

Enter these indicators in the first tab position of the row you want to alter, insert, or delete. You then enter the actual text of the prompter beginning in the second tab position.

You are now ready to begin entering the prompter data. Use the TAB key to move the cursor to the beginning of the first line, then enter the data as shown:

```
PRD003  Page = 1                MANTIS Prompter Design Facility          Lines =
.....|.....1.....|.....2.....|.../3....|....4....|....5....|../6....|....7....|..
i al - alabama/ky - kentucky/nd - north dakota
i ak - alaska/la - louisiana/oh - ohio
i az - arizona/me - maine/ok - oklahoma
i ar - arkansas/md - maryland/or - oregon
i ca - california/ma - massachusetts/pa - pennsylvania
i co - colorado/mi - michigan/ri - rhode island
i ct - connecticut/mn - minnesota/sc - south carolina
i de - delaware/ms - mississippi/sd - south dakota
i dc - district of columbia/mo - missouri/tn - tennessee
i fl - florida/mt - montana/tx - texas
i ga - georgia/ne - nebraska/ut - utah
i hi - hawaii/mv - nevada/vt - vermont
i id - idaho/nh - new hampshire/va - virginia
i il - illinois/nj - new jersey/wa - washington
i in - indiana/nm - new mexico/wv - west virginia
i ia - iowa/ny - new york/wi - wisconsin
i ks - kansas/nc - north carolina/wy - wyoming
```

As you enter the state data, experiment with the A (ALTER) and D (DELETE) commands, as well as with I (INSERT).



After you have entered all of the data, press ENTER. MANTIS evaluates your entries and displays the prompter according to the tabs you specified:

```

PRD003  Page = 1                MANTIS Prompter Design Facility          Lines =17
....|....1....|....2....|.../3....|....4....|....5....|.../6....|....7....|..
AL - ALABAMA                    KY - KENTUCKY                        ND - NORTH DAKOTA
AK - ALASKA                     LA - LOUISIANA                    OH - OHIO
AZ - ARIZONA                    ME - MAINE                        OK - OKLAHOMA
AR - ARKANSAS                   MD - MARYLAND                     OR - OREGON
CA - CALIFORNIA                 MA - MASSACHUTSETTS              PA - PENNSYLVANIA
CO - COLORADO                   MI - MICHIGAN                    RI - RHODE ISLAND
CT - CONNECTICUT               MN - MINNESOTA                   SC - SOUTH CAROLINA
DE - DELAWARE                   MS - MISSISSIPPI                 SD - SOUTH DAKOTA
DC - DISTRICT OF COLUMBIA      MO - MISSOURI                    TN - TENNESSEE
FL - FLORIDA                    MT - MONTANA                      TX - TEXAS
GA - GEORGIA                    NE - NEBRASKA                    UT - UTAH
HI - HAWAII                     NV - NEVADA                       VT - VERMONT
ID - IDAHO                      NH - NEW HAMPSHIRE                VA - VIRGINIA
IL - ILLINOIS                   NJ - NEW JERSEY                  WA - WASHINGTON
IN - INDIANA                    NM - NEW MEXICO                  WV - WEST VIRGINIA
IA - IOWA                       NY - NEW YORK                     WI - WISCONSIN
KS - KANSAS                      NC - NORTH CAROLINA              WY - WYOMING

```

Notice that wherever you designated a tab character, MANTIS moves the data directly following that tab character to the next tab position.

MANTIS also supplies the current page number (you can have up to four pages in a prompter), and the number of lines displayed on the current page. If your prompter has entries filling more than one page, you can page through them by entering the page number after PAGE, or by using the PF1–PF4 keys.

Press the CANCEL key to return to the Prompter Design Facility menu.

### Step 3: Saving the prompter

You must now save the prompter design. Select the Library functions option from the Prompter Design Facility menu by entering 3 in the action field and pressing ENTER. MANTIS displays the Prompter Design Library Facility menu:

PRD005

MANTIS

Prompter Design Library Facility

Name of prompter ..... :

Password ..... :

Chain to next prompter ..... :

:

Save ..... 1

Replace ..... 2

Fetch ..... 3

Delete ..... 4

Terminate ..... CANCEL

:

If you have an existing prompter design in your work area, the name and description of that prompter displays. (The password does not display.)

Since this is a new prompter, you must supply a name, description, and password to save the design. Enter the information as shown (continue to use your user ID as the password, although it will not display as you enter it):

```

PRD005                                M A N T I S

                                Prompter Design Library Facility

Name of prompter ..... : state_codes      :
Password .....         : password         :
Chain to next prompter ..... :           :
: state codes           :                 :

Save ..... 1
Replace ..... 2
Fetch ..... 3
Delete ..... 4
Terminate ..... CANCEL

                                : 1 :

```

You can use the Chain to next prompter field to make the prompter longer than four pages. (The additional prompter can be named here and created later.) However, for the state codes prompter, leave this field blank.

The description you provide will become the centered title for the prompter when the final design is displayed. It will also appear in your directory.

Save the prompter by entering 1 in the action field and pressing ENTER, or by pressing PF1. MANTIS automatically returns you to the Prompter Design Facility menu and displays a confirmation message in the lower, left corner of the screen.

**Step 4: Viewing the directory of prompters**

Use the Directory of prompters option to display an alphabetic listing of all existing prompters.

To display the list, select the Directory of prompters option from the Prompter Design Facility menu by entering 4 in the action field and pressing ENTER (or by pressing PF4). MANTIS displays the Directory of Prompters screen:

DIR002		Directory of Prompters		YYYY/MM/DD
BURRYS				HH:MM:SS
-----Name-----	-----Password-----	-----Description-----		
STATE_CODE	PASSWORD	STATE CODES		

Press ENTER to return to the Prompter Design Facility menu.

## Step 5: Displaying the completed prompter design

You can check the final prompter design by selecting the Display completed design option. On the Prompter Design Facility menu, enter 5 in the action field and press ENTER. MANTIS displays the prompter currently in your work area:

STATE CODES		
AL - ALABAMA	KY - KENTUCKY	ND - NORTH DAKOTA
AK - ALASKA	LA - LOUISIANA	OH - OHIO
AZ - ARIZONA	ME - MAINE	OK - OKLAHOMA
AR - ARKANSAS	MD - MARYLAND	OR - OREGON
CA - CALIFORNIA	MA - MASSACHUTSETTS	PA - PENNSYLVANIA
CO - COLORADO	MI - MICHIGAN	RI - RHODE ISLAND
CT - CONNECTICUT	MN - MINNESOTA	SC - SOUTH CAROLINA
DE - DELAWARE	MS - MISSISSIPPI	SD - SOUTH DAKOTA
DC - DISTRICT OF COLUMBIA	MO - MISSOURI	TN - TENNESSEE
FL - FLORIDA	MT - MONTANA	TX - TEXAS
GA - GEORGIA	NE - NEBRASKA	UT - UTAH
HI - HAWAII	NV - NEVADA	VT - VERMONT
ID - IDAHO	NH - NEW HAMPSHIRE	VA - VIRGINIA
IL - ILLINOIS	NJ - NEW JERSEY	WA - WASHINGTON
IN - INDIANA	NM - NEW MEXICO	WV - WEST VIRGINIA
IA - IOWA	NY - NEW YORK	WI - WISCONSIN
KS - KANSAS	NC - NORTH CAROLINA	WY - WYOMING

Your completed prompter design should correspond with the one above. Notice that the description ("STATE CODES") has become the title for the prompter.

You can return to the Prompter Design Facility menu at any time during the prompter design phase and select the Display completed design option to view the prompter design.

Press ENTER to move through multiple pages of the prompter display. When you are finished, press ENTER or CANCEL to end the prompter display and return to the Prompter Design Facility menu.

## **Step 6: Printing the completed prompter design**

If you would like to view a printed copy of the prompter design, select the Print completed design option. (Enter 6 in the action field and press ENTER.) The printout will be routed to your designated printer.

---

## **Exercise**

Since the state codes prompter you just created is the only prompter for the Burrys scenario, there are no exercises for this chapter.

You are now ready to proceed to chapter 5, where you will learn MANTIS programming fundamentals.

# 5

## Understanding MANTIS programming fundamentals

Before you begin working on the Burrys menu program, you'll need to learn some basic features of the MANTIS programming language and the MANTIS Program Design Facility. This chapter introduces you to these concepts, then guides you step-by-step through the process of creating the menu program for the Burrys application.

### Learning outline

In this chapter, you will learn how to perform the following:

- ◆ Understand basic conventions of the MANTIS language, including the number set, character set, symbolic names, numeric and text expressions, and program comments.
- ◆ Access the Program Design Facility menu and select options on it.
- ◆ Use the EDIT Program Entry screen to specify a name and description for your program.
- ◆ Use the Full-Screen Editor to enter program code and save your program.
- ◆ Terminate the Full-Screen Editor.

## Basic concepts: Understanding MANTIS language conventions

Many conventions of the MANTIS language are discussed throughout the programming lessons in this tutorial. However, before you begin to write a program, you should review the basic conventions presented in the following sections.

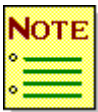
### MANTIS number set

The MANTIS number set consists of:

- ◆ Digits 0–9
- ◆ Preceding plus or minus sign
- ◆ Period (decimal point)
- ◆ Letter E

Internally, MANTIS stores numeric data in floating point (scientific notation) and regards numbers in one of two ways:

- ◆ **SMALL.** Stores a four-byte floating point number. Use SMALL to represent integers with magnitude up to 16,000,000 (approximately  $1e7$ ).
- ◆ **BIG.** Stores an eight-byte floating point number. Use BIG to represent:
  - Numbers with decimal fractions
  - Large numbers



---

Even if your installation uses a decimal point character other than the period (.) in user screens (for example, the comma), you must use a period for a decimal point in the numbers in your programs.

---



## MANTIS character set

The MANTIS character set consists of:

- ◆ Alphabetic characters A–Z
- ◆ Space character
- ◆ Numeric digits 0–9
- ◆ Special characters, such as the underline (\_) and vertical bar (|).

## Understanding literals, symbolic names, variables, and expressions

Before you begin creating MANTIS programs, you should understand the following terms:

- ◆ **Literal.** A direct representation of a value (for example, 4 or “ACCOUNT”).

In MANTIS, a numeric literal can consist of the following:

- Numbers 0 through 9
- E (exponent for scientific notation, or “e-notation”)
- Plus sign (+)
- Minus sign (-)

In MANTIS, a text literal can consist of a quoted string of any valid EBCDIC characters.

- ◆ **Symbolic name.** A string of characters that represents a user-defined object in a MANTIS program (for example, MAP, CREDIT\_LIMIT). MANTIS uses symbolic names to represent variables processed by MANTIS programs. A symbolic name can represent a complex entity such as a screen or file, or it can represent a text or num field.
- ◆ **Variable.** A symbolic name that contains a value. Variables can have different values at different points in time (for example, X, where X can be any number). Variables can represent:
  - A single value (a scalar)
  - A text array
  - A numeric array of values
- ◆ **Expression.** A combination of text or numeric variables, functions, and/or literals that can be evaluated by MANTIS.
- ◆ **Reserved word.** A word that cannot be used for user variable (symbolic) names. MANTIS uses reserved words for statements, functions, commands, and connector words. Examples of reserved words include FILE, NOT, EDIT, and TO.

## Using symbolic names

Symbolic names can represent either numeric or text data. MANTIS allows a maximum of 2048 symbolic names for a single program, including names defined indirectly by SCREEN, FILE, and ACCESS statements.

A symbolic name:

- ◆ Must begin with an alphabetic character.
- ◆ Can contain alphabetic characters, numeric characters, and the underline (\_). *No spaces or other special characters are allowed.*
- ◆ Can be entered as lowercase letters; MANTIS will convert them to uppercase (for example, the following symbolic names are equivalent: customer\_name, Customer\_Name, and CUSTOMER\_NAME).
- ◆ Can be any size that fits on a line. However, if the field is used in a design entity (that is, screens, interfaces, or files), the symbolic name is limited to 16 or 30 characters, depending upon the entity.
- ◆ Must be unique. If MANTIS encounters a definition for a symbolic name for the second time (a duplicate definition), it ignores the second definition and does not create a second work area for that name.
- ◆ Can contain a reserved word (for example, EDITOR), but cannot be a reserved word in its entirety (for example, EDIT).



---

Do not use hyphens to connect words in symbolic names. If you do, MANTIS will try to subtract the value of the second word from the first.

---

The following table shows examples of symbolic names, indicating whether each is valid or invalid:

Symbolic name	Valid/invalid
INVESTMENT	Valid
COST_OF_SALES	Valid
SALES_2000	Valid
_SALES_2000	Invalid; must begin with alphabetic character
2000_SALES	Invalid; must begin with alphabetic character
PROGRAM	Invalid; PROGRAM is a reserved word
MENU_PROGRAM	Valid
CUST NAME	Invalid; must not contain blank spaces
CREDIT-LIMIT	Invalid; must not contain a hyphen

## Using literals, variables, and expressions

MANTIS supports numeric and text literals, variables, and expressions as described in the following sections.

### Numeric literals, variables, and expressions

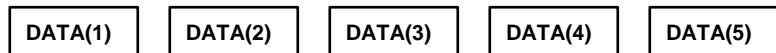
*Numeric literals* consist of the digits 0 through 9 and can have the unary signs plus and minus (for example, -9). Numbers may optionally contain one decimal point (for example, 3.14). Numbers may also use the e-notation (exponential). For example, 6.02257E+23 means 6.02257 times 10 to the 23<sup>rd</sup> power.

*Numeric variables* consist of a valid MANTIS symbolic name. A numeric variable can represent a single entity (for example SUM, where SUM contains a number) or a numeric array.

MANTIS can store numeric values in ordered sets called *arrays*. Arrays can have one or two dimensions. You can specify arrays as BIG or SMALL. For example, if you specify a one-dimensional array as follows:

```
BIG DATA(5)
```

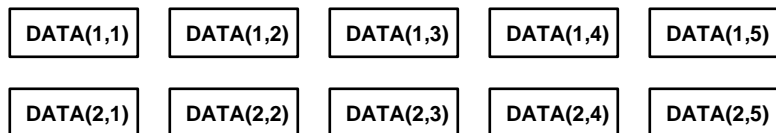
MANTIS allocates storage for five BIG occurrences of DATA:



If you specify a two-dimensional array of two rows and five columns:

```
BIG DATA(2,5)
```

MANTIS allocates storage as a two-dimensional array:



If the variable represents an array, individual elements can be referenced by subscripting the variable name, for example MONTHLY\_SALES(3) represents the value for the third element of the array.

*Numeric expressions* consist of a combination of numeric variables and/or literals (for example, FINAL\_SUM=SUM+9).

## Text literals, variables, and expressions

A *text literal* is any set of 0-254 EBCDIC characters enclosed in quotes (for example, "THIS IS A TEXT LITERAL"). Use two consecutive quotes to specify one occurrence of a quotation mark within a text literal. For example, if you issue the following command:

```
SHOW "THE ANSWER IS "Y" "
```

MANTIS returns the following:

```
THE ANSWER IS "Y"
```

A *text variable* is a symbolic name that can assume up to 254 EBCDIC characters (letters, numerals, punctuation, native language characters, etc.). The default for a variable is numeric (see BIG), so you *must* define text variables before using them. If you specify:

```
TEXT DATA(5)
```

MANTIS allocates 5 characters of memory for the variable, DATA. You can also define a text array in your program. For example, if you enter the statement:

```
TEXT DATA(3,10)
```

MANTIS allocates storage as:

DATA(1)
---------

DATA(2)
---------

DATA(3)
---------

where each DATA element has 10 bytes (characters) of storage available.

*Text expressions* consist of a combination of text variables and/or literals (for example, ACCOUNT\_INFO="ACCOUNT "+ACCOUNT\_NUMBER).

## **Defining variables in a program**

You can define a variable in a program in the following ways:

- ◆ With variable definitions within a SCREEN, FILE, INTERFACE, TOTAL, ACCESS, or VIEW layout. The variable becomes defined when the statement associated with the complex entity is executed. (If text, the variable is initially a zero-length string; if numeric, the variable is initially zero.)
- ◆ With a TEXT, BIG, SMALL, or KANJI statement. (If text, the variable is initially a zero-length string; if numeric, the variable is initially zero.)
- ◆ As a parameter on the program's main entry statement where the argument was passed. (The variable has the definition and the actual data of the invoking program's parameter.)
- ◆ With an as-yet undefined symbolic name in a statement. It will default to a BIG variable with a value of zero.

# Step-by-step: Creating the customer menu program

This section provides a step-by-step introduction to MANTIS program design, including:

- ◆ Accessing the Program Design Facility
- ◆ Using the EDIT Program Entry screen
- ◆ Creating a program in the Full-Screen Editor (FSE)
- ◆ Terminating the Full-Screen Editor

MANTIS programs are created and managed in *programming mode*. To enter MANTIS programming mode, select the Design a Program option from the MANTIS Facility Selection menu. The Program Design Facility menu displays:

```
PRGMENU01      Program Design Facility (BURRYS)      YYYY/MM/DD HH:MM:SS
===>

Please select one of the menu items below.

      Program      Component Engineering  Bind Options  Utilities
--  1. List        7. CEF Check          12. HPO Check  18. Audit Trail
    2. Edit        8. " Compose          13. " Bind     19. Browse Audit Trail
    3. Profile     9. " Decompose         14. " Unbind   20. " Prgm Profile
    4. Purge       10. CREF Programs       15. SQL Check  21. Trigger List
    5. Copy        11. Bill of Materials   16. " Bind     22. SQL Maint
    6. Rename                        17. " Unbind

FAC000I:  READY
F1=HELP  F2=EXHELP  F3=EXIT  F4=PROMPT  F5=REFRESH  F9=RETRIEVE F12=CANCEL ...
```

From the Program Design Facility menu, you can select options under these group headings: Program, Component Engineering, Bind Options, and Utilities. To create the Burrys menu program, you will use options from the Program group to access the Full-Screen Editor and perform other tasks related to creating and maintaining MANTIS programs.



## Step 1: Using the EDIT Program Entry screen

From the Program Design Facility menu, enter the EDIT command on the command line (==>):

```
PRGMMENU01      Program Design Facility (BURRYS)      YYYY/MM/DD HH:MM:SS
==> edit

Please select one of the menu items below.

      Program      Component Engineering      Bind Options      Utilities
--  1. List          7. CEF Check              12. HPO Check      18. Audit Trail
    2. Edit          8.  "  Compose              13.  "  Bind        19. Browse Audit Trail
    3. Profile       9.  "  Decompose             14.  "  Unbind       20.  "  Prgm Profile
    4. Purge        10. CREF Programs             15. SQL Check       21. Trigger List
    5. Copy         11. Bill of Materials          16.  "  Bind        22. SQL Maint
    6. Rename                          17.  "  Unbind

FAC000I:  READY
F1=HELP  F2=EXHELP  F3=EXIT  F4=PROMPT  F5=REFRESH  F9=RETRIEVE F12=CANCEL ...
```

Press ENTER. The EDIT Program Entry screen displays:

```
PRGMENT101E      EDIT Program Entry      YYYY/MM/DD HH:MM:SS
==>
From
  Library . . .   BURRYS
  Name . . .
  Description .

Thru
  Name . . .

Entry Options      Function Options      Process Statistics
Immediate? . . . Y  Uppercase? . . . . Y    Processed . .
Confirmation? . . N Nulls on? . . . . Y    Skipped . . .
                                Indent on? . . . . Y    Errors . . . .
                                Scroll? (P H C). . . P

000:  READY
F1=HELP  F2=EXHELP  F3=EXIT  F4=PROMPT  F5=REFRESH  F6=EXECUTE  F7=CONFIRM ...
```

This screen lets you designate a new program name and specify Entry and Function options that will be in effect for the duration of your session.

Entry Options and Function Options on the EDIT Program Entry panel control the way the Edit action is executed from this panel. You can change the settings of these options, but when you exit from the Full-Screen Editor the changed settings default to their original values. For now, we will accept the default settings for these options.

To access the Full-Screen Editor to create the Burrys menu program:

1. Enter the program name CUST\_MENU in the From Name field.
2. Enter BURRYS CUSTOMER ACCOUNTS MENU in the description field. Your screen should look like this:

```

PRGMENT101E      EDIT Program Entry      YYYY/MM/DD HH:MM:SS
==>
From
  Library . . .   BURRYS
  Name . . . .   cust_menu                Password :           :
  Description .   burrys customer accounts menu

Thru
  Name . . . .

Entry Options      Function Options      Process Statistics
Immediate? . . . . Y      Uppercase? . . . . . Y      Processed . .
Confirmation? . . N      Nulls on? . . . . . Y      Skipped . . .
                        Indent on? . . . . . Y      Errors . . . .
                        Scroll? (P H C). . . P

000: READY
F1=HELP F2=EXHELP F3=EXIT F4=PROMPT F5=REFRESH F6=EXECUTE F7=CONFIRM ...

```

3. Leaving the remaining fields blank, press the EXECUTE key, PF6 (or enter the EXECUTE command on the command line and press ENTER), to display an empty FSE work area.

Once you enter FSE, you can create, modify, and execute MANTIS programs.



To create the new menu program:

1. Enter the following program statements next to the single quotes (located in the Line Number field). *Do not enter line numbers:*

```
ENTRY CUST_MENU
SCREEN MAP( "CUST_MENU" )
CONVERSE MAP
WHILE MAP<>"CANCEL"
WHEN ACTION=1 OR MAP="PF1"
SHOW"PROGRAM CHAINS TO CUSTOMER ENTRY SCREEN"
WAIT
WHEN ACTION=2 OR MAP="PF2"
SHOW"PROGRAM CHAINS TO CUSTOMER BROWSE SCREEN":WAIT
END
CLEAR MAP
CONVERSE MAP
END
EXIT
```

Use the arrow or TAB keys to move the cursor from one line to the next. (The function of each statement will be explained in the next chapter.) When you've finished entering the program lines, your screen should look like this:

```
EDIT L1--- BURRYS:CUST_MENU                                COLUMNS 1      73
COMMAND ==>>                                              SCROLL ==>    PAGE
***** ***** START OF PROGRAM *****
| | | | entry cust_menu
| | | | screen map("cust_menu")
| | | | converse map
| | | | while map<>"cancel"
| | | | when action=1 or map="pf1"
| | | | show "program chains to customer entry screen"
| | | | wait
| | | | when action=2 or map="pf2"
| | | | show "program chains to customer browse screen":wait
| | | | end
| | | | clear map
| | | | converse map
| | | | end
| | | | exit
| | | |
| | | |
***** ***** END OF PROGRAM *****
```

2. Press ENTER. MANTIS assigns line numbers to the lines you entered and deletes the extra blank lines and unneeded blank spaces on the panel. MANTIS also indents the program lines to indicate the logical structure of the program. (You could also have inserted one line at a time from the command line by entering a new line number and text, then pressing ENTER to add the line to your program.)
3. At the FSE command line (===>), enter the following command; then, press ENTER:

```
SEQUENCE 100,10
```

MANTIS rennumbers your program line numbers, beginning at line 100 and incrementing the line numbers by ten.

4. Enter SAVE at the FSE command line and press ENTER. The SAVE command saves the new program in your library.

You have just created and saved your first MANTIS program! In the next chapter, you'll take a closer look at this program, examining the code step-by-step to learn the purpose of each line. For now, just get a general sense of how a MANTIS program is constructed. You'll also learn how to use FSE primary and line commands to edit your program and save your changes.

### Step 3: Terminating the Full-Screen Editor

There are several ways to terminate the Full-Screen Editor:

- ◆ The END command saves or replaces the program and terminates the Full-Screen Editor, returning you to the Facility Selection screen.
- ◆ The QUIT command terminates the Full-Screen Editor, returning you to the Program Design facility menu, *without saving or replacing the program*. Be sure to SAVE or REPLACE your program before using the QUIT command because MANTIS does not ask you to confirm your exit. (You can also use the QUIT command to abandon the changes you've made since the last REPLACE or SAVE.)
- ◆ Entering CANCEL (or CAN) at the command line or pressing the CANCEL key terminates the Full-Screen Editor, returning you to the Program Design Facility menu. If you modified your program, MANTIS asks for confirmation of your exit; if you did not modify your program, MANTIS does not ask for confirmation.
- ◆ The LOGOFF command saves or replaces the program and terminates MANTIS, returning you to CICS.

Since you have already saved your program, press the CANCEL key (or enter CANCEL at the command line and press ENTER) to terminate the Full-Screen Editor. The EDIT Program Entry screen displays, indicating the range of programs (in this instance the beginning and ending programs are the same) that were processed.

To return directly to the MANTIS Facility Selection menu (bypassing the Program Design Facility menu), enter MENU on the command line and press ENTER.

---

## Exercises

Complete the exercises in this section before you proceed to the next chapter.

### Exercise 1: Verifying the menu program

Examine the menu program that you just created and make sure that it's identical to this one:

```
00100 ENTRY CUST_MENU
00110 .SCREEN MAP( "CUST_MENU" )
00120 .CONVERSE MAP
00130 .WHILE MAP<>"CANCEL"
00140 ..WHEN ACTION=1 OR MAP="PF1"
00150 ...SHOW"PROGRAM CHAINS TO CUSTOMER ENTRY SCREEN"
00160 ...WAIT
00170 ..WHEN ACTION=2 OR MAP="PF2"
00180 ...SHOW"PROGRAM CHAINS TO CUSTOMER BROWSE SCREEN":WAIT
00190 ..END
00200 ..CLEAR MAP
00210 ..CONVERSE MAP
00220 .END
00230 EXIT
```

If your program is not identical to the one above, update your program by typing over the existing program lines. (Press DELETE to remove any extra characters.)

If you need to add a program line, enter the line number and program statement on the FSE command line (==>); then, press ENTER to add the line to your program. To delete an extra line from your program, enter d on the number of the line you want to delete; then, press ENTER.

When you are finished, save your changes by entering REPLACE on the command line and pressing ENTER.

## Exercise 2: Adding records to the state codes file

Now, to apply what we've discussed so far in this tutorial, let's create another simple program, run it, and enter records into a file.

To begin, you will need to clear the FSE work area so that you can enter a new program. To do so, enter NEW on the FSE command line (==>) and press ENTER. MANTIS clears your work area and displays a set of empty lines where you can enter your new program statements.

Follow the procedures discussed in this chapter to create and save the following program:

```
00100 ENTRY STATE_CODES
00110 .SCREEN MAP( "STATE_CODE" )
00120 .FILE REC( "STATE_CODES" , "PASSWORD" )
00130 .CONVERSE MAP
00140 .WHILE MAP<>"CANCEL"
00150 ..INSERT REC
00160 ..CLEAR MAP
00170 ..CONVERSE MAP
00180 .END
00190 EXIT
```



---

Remember, do not enter the line numbers in the FSE work area when you enter your program statements. To save your program, enter SAVE on the Full-Screen Editor command line; then, press ENTER.

---

Now, enter RUN on the command line and press ENTER. MANTIS displays the state code entry screen (STATE\_CODE) that you created in chapter 2.

In the CUST\_STATE field, enter a state code abbreviation and press ENTER. MANTIS adds the state code to the STATE\_CODES file. Add four or five more state codes to the file. (We'll use them in a later exercise.) When you're finished, press the CANCEL key to terminate the program and return to the Full-Screen Editor.



### Exercise 3: Adding records to the customer information file

Now, let's add a few records to the customer information file, so that we can use them in a future exercise. To do so, we'll create another program, run it, and enter records into the file.

Use the procedures discussed in this chapter to create and save the following program:

```
00100 ENTRY ADD_CUST
00110 .SCREEN MAP( "CUST_ENTRY" )
00120 .FILE REC( "CUST_INFO" , "PASSWORD" )
00130 .CONVERSE MAP
00140 .WHILE MAP<>"CANCEL"
00150 ..INSERT REC
00160 ..CLEAR MAP
00170 ..CONVERSE MAP
00180 .END
00190 EXIT
```

Next, enter RUN on the command line and press ENTER. MANTIS displays the customer entry screen (CUST\_ENTRY) that you created in chapter 2.

Enter a customer record; then, press ENTER. MANTIS adds the customer record to the CUST\_ENTRY file. (Because MANTIS automatically maps data between like-named fields on the screen and file record, no program code is required to move the data.) Add several more customer records to the file; then, press the CANCEL key to terminate the program and return to the Full-Screen Editor.

Press PF3 to save the program and exit from the Full-Screen Editor.

These exercises demonstrate how easily and quickly you can create applications using MANTIS. In the following chapters, you'll learn more about MANTIS programming statements and commands, and how to use them to build more complex programs.



# 6

## Using MANTIS programming statements and commands

Now that you know how to access the Full-Screen Editor (FSE) and how to create a program, it's time to learn more about the MANTIS programming environment. This chapter provides an overview of all the FSE commands, and shows you how to use them to modify the menu program that you created in the previous chapter.

### Learning outline

In this chapter, you will:

- ◆ Understand the difference between programming statements and immediate mode statements.
- ◆ Understand how to use basic MANTIS programming statements.
- ◆ Understand MANTIS commands and use them to edit a program in the Full-Screen Editor.
- ◆ Learn how to create program stubs for the CUST\_ENTRY and CUST\_BROWSE programs.

## Basic concepts: Understanding statements and commands

The MANTIS programming environment is comprised of these elements:

- ◆ **Statements.** There are two types of MANTIS statements:
  - Program statements—A *program statement* consists of a line number, a MANTIS language element, and possibly one or more operands. As part of a program, program statements require a run-mode action; that is, they are not executed until the program is run.

MANTIS automatically indents program statements to indicate their relative position in the program's nesting hierarchy. The following line shows an example of a program statement:

```
300 FILE REC("MYFILE", "MYFILEPSWD")
```

- Immediate mode statements—An *immediate mode statement* is one that is entered without a line number, on the command line of the Full-Screen Editor, indicating that it should be interpreted and executed immediately and not become a part of the program.

While an immediate mode statement uses statements that are also valid in a MANTIS program, not all program statements can be entered in immediate mode. (For example, logic statements such as IF, WHEN, and WHILE are not valid.) So, if entered on the FSE command line, the following statement will be executed immediately and is not part of the saved program:

```
FILE REC("MYFILE", "MYFILEPSWD")
```

- ◆ **Commands.** A *command* is not part of a program, but is an order that you give to MANTIS when you want it to perform some action (such as SAVE, QUIT, EDIT, etc.). The Full-Screen Editor uses editing commands to maintain programs. A command designates an immediate mode action, and is executed immediately. A command cannot have a line number.

Some MANTIS reserved words can be used as both statements and commands (such as EXIT).

## Using MANTIS program statements

To gain a better understanding of MANTIS program statements, let's take a closer look at the Burrys menu program that you created in chapter 5:

```
00100 ENTRY CUST_MENU
00110 .SCREEN MAP( "CUST_MENU" )
00120 .CONVERSE MAP
00130 .WHILE MAP<>"CANCEL"
00140 ..WHEN ACTION=1 OR MAP="PF1"
00150 ...SHOW"PROGRAM CHAINS TO CUSTOMER ENTRY SCREEN"
00160 ...WAIT
00170 ..WHEN ACTION=2 OR MAP="PF2"
00180 ...SHOW"PROGRAM CHAINS TO CUSTOMER BROWSE SCREEN":WAIT
00190 ..END
00200 ..CLEAR MAP
00210 ..CONVERSE MAP
00220 .END
00230 EXIT
```

### ENTRY-EXIT statements

The ENTRY and EXIT statements define the boundaries of your program:

```
00100 ENTRY CUST_MENU
...
00230 EXIT
```

Use the ENTRY-EXIT statements around programs. A symbolic name (in this case, CUST\_MENU) must follow ENTRY. Remember that a MANTIS symbolic name must begin with a letter and must use the underscore to connect parts of the name. (Do not use blanks to connect parts of the name, or MANTIS will treat the parts as separate symbolic variables.)

## SCREEN statement

In line 110, the menu program specifies the screen to be used:

```
110 .SCREEN MAP( "CUST_MENU" )
```

As part of the SCREEN statement, you must supply a symbolic name (in this case, MAP) as well as the name you gave your screen when you saved it ("CUST\_MENU"). Enclose the screen name in quotes and parentheses.



---

Some MANTIS sites implement standard naming conventions that require the entity name to be prefixed with the library name (that is, the user ID name) in all program statements that specify MANTIS entities (SCREEN, FILE, etc.). This practice allows the program to be copied to, and run from, another MANTIS user while the other entities reside on the original user.

To include the library name, use the syntax "library:entity\_name". For example, using this syntax the preceding SCREEN statement would read:

```
110 .SCREEN MAP( "BURRYS:CUST_MENU" ) .
```

---

Although the SCREEN statement identifies a screen to your program and assigns it a symbolic name, the SCREEN statement does not actually display the screen.

## CONVERSE statement

In line 120, the program displays the screen that you specified in the SCREEN statement:

```
120 .CONVERSE MAP
```

The CONVERSE statement tells the program to display a particular screen. Use the CONVERSE statement with the screen's *symbolic* name. The CONVERSE statement also moves any user input from the terminal into the defined screen variables.

When you execute the menu program, CONVERSE sends the screen to the terminal. You can then select one of the options on the screen (1, 2, or CANCEL), and MANTIS returns your response to the program, either in your screen variable (ACTION) or the symbolic name of the screen (MAP) if a PF or PA key is pressed.

## WHILE-END and WHEN-END

Two basic structures control the flow of the menu program: a WHILE loop and a WHEN conditional. Loops and conditionals form the building blocks for MANTIS programs.

The WHILE loop, defined by WHILE-END statements, executes a block of statements repeatedly as long as the specified condition is true:

```
130 .WHILE MAP<>"CANCEL"  
...  
220 .END
```

The symbol <> is a MANTIS operator that means “does not equal”. Thus, your program will execute any statements that you place between lines 130 and 220, until the user presses CANCEL (the key value placed in the MAP symbolic name variable).

In the menu program, the WHILE test essentially asks the question, “You haven’t pressed CANCEL, have you?” As long as MANTIS receives the answer “no”, it executes statement 140:

```
140 ..WHEN ACTION=1 OR MAP="PF1"  
...  
170 ..WHEN ACTION=2 OR MAP="PF2"  
...  
190 ..END
```

The WHEN conditional statements in the menu program provide for the two remaining options on your menu. Remember that the data field on the menu screen is named ACTION, and that it’s a numeric field. You want the user to be able to select an option by keying in 1 or 2 and pressing ENTER, or pressing the corresponding PF key.

WHEN-END executes a block of statements only when a condition you specify is true. In the menu program, MANTIS asks at statement 140 whether or not you entered 1 or pressed PF1. If so, MANTIS executes lines 150 and 160 and then performs the next WHEN test at 170; otherwise, it performs the next WHEN test at 170. You can include any number of WHEN tests in a series. Note also that the WHEN conditional structure is not terminated when the first WHEN condition is met; all of them are checked.

## SHOW and WAIT statements

The program contains SHOW and WAIT statements that we used as a place-holder for the actual statements that we want to insert there.

The SHOW statements (lines 150 and 180) display data that you specify on an unformatted screen. For example, the SHOW statement in line 150 displays the text string "PROGRAM CHAINS TO CUSTOMER ENTRY SCREEN":

```
150 ...SHOW"PROGRAM CHAINS TO CUSTOMER ENTRY SCREEN"  
160 ...WAIT  
...  
180 ...SHOW"PROGRAM CHAINS TO CUSTOMER BROWSE SCREEN":WAIT
```

The colon (:) on line 180 tells MANTIS that another statement (in this case, WAIT) appears directly following it. So, line 180 performs an action similar to that of lines 150 and 160 combined.



---

You cannot combine every statement with another statement on a line. For example, the ENTRY, EXIT, WHEN, WHILE, and END statements must all be on lines by themselves.

---

A WAIT statement temporarily suspends execution of your program by sending any pending SHOW data to the screen and waiting for an operator response. In this example, if you enter the RUN command on the FSE command line and press ENTER, and then when the CUST\_MENU screen displays press PF1 (or enter 1 and press ENTER), your program will display the first text string and pause:

```
PROGRAM CHAINS TO CUSTOMER ENTRY SCREEN
```

If you had pressed PF2 instead, the other text string would have displayed:

```
PROGRAM CHAINS TO CUSTOMER BROWSE SCREEN
```

Press ENTER to return to the CUST\_MENU screen; then, press the CANCEL key to return to the FSE program work area.



---

You can use SHOW and WAIT statements as a debugging tool, by placing them at strategic locations throughout a MANTIS program.

---



## CLEAR statement

After an entry other than CANCEL, the menu program executes the CLEAR statement to erase the data in the ACTION field (line 200). The CLEAR statement also erases the value in the symbolic name MAP, if a PF key was pressed:

```
200 ..CLEAR MAP
210 ..CONVERSE MAP
```

Supply the screen's symbolic name (in this case, MAP) after CLEAR, to tell MANTIS which screen to clear.

The menu program follows the CLEAR statement with another CONVERSE statement to display the screen and await the next entry in the WHILE loop.

## Including comments in a program

You can include comments in a program by entering a vertical bar followed by descriptive text:

```
|THIS IS A SAMPLE COMMENT.
```

MANTIS ignores comments during program execution, so do not place program statements after the vertical bar. The one exception to this rule is that MANTIS does recognize SQL database access statements after the comment bar, when they are enclosed within the EXEC\_SQL and END statements. For more detailed information on coding the EXEC\_SQL-END structure, refer to *MANTIS Language, OS/390, VSE/ESA*, P39-5002.

## Using MANTIS programming commands

A MANTIS programming command is a reserved word *without* a line number. MANTIS does not consider a command to be part of your program and executes it immediately. For example, you use commands to SEQUENCE or RUN a program. (For a detailed explanation of the syntax of MANTIS programming commands, refer to *MANTIS Program Design and Editing, OS/390, VSE/ESA*, P39-5013.)

Programming commands can be Full-Screen Editor commands or program statements that are executed as commands.

### Using Full-Screen Editor commands

The Full-Screen Editor provides two types of commands that you can use to edit programs:

- ◆ **Primary commands.** Primary commands are entered on the command line of the Full-Screen Editor, and include global editing commands (FIND, SAVE, REPLACE, etc.).
- ◆ **Line commands.** Line commands provide editing instructions for specific lines in the Full-Screen Editor, and affect the lines on which they appear. Line commands include editing commands (MOVE, COPY, etc.) as well as destination commands (AFTER, BEFORE, etc.).

Enter commands in either uppercase or lowercase letters. MANTIS automatically translates the characters that you enter into uppercase format unless configured by the Master User to not do so.

Full-Screen Editor commands are processed in the following order:

1. Line commands are executed first, in this order: DELETE, SELECT, REPEAT, COPY, MOVE, INSERT. (The SELECT line command is a special line command that cannot be executed in combination with a primary command.)
2. Primary commands are executed after line commands.
3. Primary commands issued from PF keys (for example, scrolling, END, REPLACE) are executed last.

This table provides a quick reference for the Full-Screen Editor commands. Commands are arranged in alphabetic order, along with a brief description. The Type column indicates whether the command is an FSE primary command (P) or an FSE line command (L):

Command	Type	Description
A (after)	FSE (L)	Used with the COPY or MOVE command to indicate destination— <i>after</i> this line
B (before)	FSE (L)	Used with the COPY or MOVE command to indicate destination— <i>before</i> this line
BIND	FSE (P)	Converts a program from unbound to bound format (BIND [ON]), or from bound to unbound format (BIND OFF)
BOTTOM	FSE (P)	Scrolls the terminal window to the end of your program
C (copy)	FSE (L)	Specifies the line(s) you want to copy
CANCEL	FSE (P)	Terminates FSE without saving the program (MANTIS prompts for confirmation)
CAPS ON/OFF	FSE (P)	Specifies whether the data from the display will be treated as uppercase or lowercase (mixed)
COPY	FSE (P)	Specifies that MANTIS should copy all or part of a MANTIS program into the program being edited
D (delete)	FSE (L)	Specifies the line(s) that you want to delete
DOWN	FSE (P)	Scrolls the editor screen down
END	FSE (P)	Saves the program and terminates programming mode
ERASE	FSE (P)	Deletes one or more program lines
ERRCODE	FSE (P)	Displays the full message text corresponding to a 3-character program syntax error code
FIND	FSE (P)	Finds and displays the next occurrence of a string in a program
HELP	FSE (P)	Displays a help prompt for an error code, a command, a reserved word, or online help for FSE
I (insert)	FSE (L)	Inserts one or more blank lines after this line
KILL	FSE (P)	Terminates a program in a loop; can be changed or disabled by the Master User
LEFT <i>n</i>	FSE (P)	Scrolls the editor screen to the left “ <i>n</i> ” columns
LIST	FSE (P)	Lists all or part of the program currently in work area
LOAD	FSE (P)	Retrieves a program from a library

Command	Type	Description
LOCATE	FSE (P)	Locates a specific line in the current program
LOGOFF	FSE (P)	Saves FSE changes and exits from MANTIS
M (move)	FSE (L)	Specifies the line(s) you want to move
MENU	FSE (P)	Saves FSE changes and displays the MANTIS Facility Selection menu
MOVE	FSE(P)	Specifies the line(s) you want to move, and where
NEW	FSE (P)	Clears the current work area so that you can enter a new program
O (overlay)	FSE (L)	Used with the COPY or the MOVE command to indicate destination— <i>over</i> (in place of) this line
PRINT	FSE(P)	Routes the current program to your designated printer
PROFILE	FSE (P)	Displays the Edit Profile
PURGE	FSE (P)	Erases the program from your library
QUIT	FSE(P)	Terminates the editing session without saving the program (MANTIS does not prompt for confirmation)
R (repeat)	FSE (L)	Specifies the line(s) you want to repeat
RCHANGE	FSE (P)	Repeats the last CHANGE command that was entered
REPLACE	FSE (P)	Replaces the program in your library with the program currently being edited
RESET	FSE (P)	Resets any pending primary commands, line commands, and commands issued from PF keys
RFIND	FSE (P)	Repeats the last FIND command entered
RIGHT <i>n</i>	FSE (P)	Scrolls the editor screen to the right “ <i>n</i> ” columns
RUN	FSE (P)	Executes the program currently in the work area
S (select)	FSE (L)	Selects component-engineered source programs and components for editing
SAVE	FSE (P)	Copies the edited program into the library
SCROLL	FSE (P)	Determines the scrolling mode
SEQUENCE	FSE (P)	Renumbers the lines of the program that you are editing
TOP	FSE (P)	Scrolls the terminal window to the top of your program
UP	FSE (P)	Scrolls the editor screen up

For more information about the format of each command and guidelines for usage, refer to *MANTIS Program Design and Editing, OS/390, VSE/ESA*, P39-5013.

## Using statements as commands

You can execute some statements as commands by entering them without a statement number. Such a statement is not part of the coded program and MANTIS executes it immediately. In the following example, **SHOW** appears as both a statement and a command. First, in this program **SHOW** appears as a *statement*:

```
10  ENTRY COMPOUND
20  .SHOW"THIS PROGRAM DEMONSTRATES THE SHOW STATEMENT."  <-statement
30  .WAIT
40  .LET INVESTMENT=1400
50  EXIT
```

If you enter the **RUN** command and press **ENTER**, MANTIS displays the following:

```
THIS PROGRAM DEMONSTRATES THE SHOW STATEMENT.
```

If you then press **ENTER** to return to FSE, you can enter the **SHOW** statement as a *command* on the FSE command line and press **ENTER**:

```
SHOW INVESTMENT+650                                <-command
```

The above command tells MANTIS to add 650 to **INVESTMENT** (which was assigned a value of 1400 in line 40 of the program), and displays the following result above the command line:

```
2050
```

You can see how using statements as commands might help you to test and debug a program. You can stop the program at various points and view or assign variables to help determine how your program is executing.

# Step-by-step: Completing the menu program

To complete the Burrys menu program, we'll add a comment and replace the SHOW and WAIT statements with CHAIN statements.

Since the MANTIS development environment is interactive and programs can be executed immediately without compiling, it's easy to develop a program in a series of steps, testing each step along the way. This process is called *step-wise refinement*, because each step adds more detail and function to the program. This manual uses step-wise refinement, iteratively developing, testing and refining each program. To introduce you to this process, some intentional mistakes have been left for you to discover and repair.

The interactive nature of the MANTIS development environment also makes it easy to demonstrate partially completed programs to end-users to ensure that the finished application will meet their needs. This process is called *prototyping*. The programs you will be developing in the following sections will apply the step-wise refinement and prototyping techniques.

To begin, select Design a Program from the Facility Selection menu. At the command line on the Program Design Facility menu, enter EDIT CUST\_MENU (or, 2 CUST\_MENU); then, press ENTER:

```
PRGMMENU01      Program Design Facility (BURRYS)                YYYY/MM/DD HH:MM:SS
===> edit cust_menu

Please select one of the menu items below.

      Program      Component Engineering  Bind Options      Utilities
__  1. List        7. CEF Check          12. HPO Check      18. Audit Trail
    2. Edit        8.  "  Compose         13.  "  Bind       19. Browse Audit Trail
    3. Profile     9.  "  Decompose        14.  "  Unbind     20.  "  Prgm Profile
    4. Purge       10. CREF Programs          15. SQL Check      21. Trigger List
    5. Copy        11. Bill of Materials      16.  "  Bind       22. SQL Maint
    6. Rename      17.  "  Unbind

FAC000I:  READY
F1=HELP  F2=EXHELP  F3=EXIT  F4=PROMPT  F5=REFRESH  F9=RETRIEVE F12=CANCEL ...
```

MANTIS opens the Full-Screen Editor and lists your program.

## Step 1: Adding a comment

First, we'll use the INSERT command (a Full-Screen Editor line command) to add a comment line after 110. The INSERT command inserts a blank line in your program. Enter "I" on the first zero in line 110:

```
i0110 .SCREEN MAP("CUST_MENU")
```

When you press ENTER, MANTIS inserts a blank line *after* line 110:

```
00110 .SCREEN MAP("CUST_MENU")
      .
      .
      .
00120 .CONVERSE MAP
```

A blank line is identified by single quotes, just as when you create a new program using the Full-Screen Editor. Next to the single quotes enter the following comment:

```
      ' ' ' ' | This is the customer accounts menu
```

Remember, comments begin with a vertical bar and include the rest of the line. When you press ENTER, MANTIS assigns a line number to your new statement and inserts another blank line. Press ENTER again to delete the extra blank line.

You can also include comments on the same line with a program statement by entering a colon (:), a vertical bar (|), and your comment after the statement:

```
10  ENTRY CUST_MENU:|THIS IS AN EXAMPLE
```



Because MANTIS ignores comments when executing a program (except for SQL statement text), don't place statements or commands on the same line after a comment.

## Step 2: Finding and changing a program line

Next, we want to change the SHOW statements to CHAIN statements. We will use the FIND command to find the SHOW statements, so that we can change them.

The FIND command finds text in your program. (F is the abbreviation for the command.) MANTIS will find the next occurrence of the text unless you specify otherwise (PREVIOUS, FIRST, LAST, ALL). You can also specify whether MANTIS should find an occurrence with the text as a suffix, prefix, or an entire word.

For now, enter F SHOW on the command line and press ENTER. MANTIS highlights the line that contains the first SHOW statement and places the cursor in the first position of the text string. A confirmation message appears in the Title/Message Field:

```

EDIT --- FAC042I:CHARS 'SHOW' FOUND                                COLUMNS 1      73
COMMAND ==>                                                         SCROLL ==>    PAGE
***** ***** START OF PROGRAM *****
00100 ENTRY CUST_MENU
.
.
.
.
00150 ..._SHOW "PROGRAM CHAINS TO CUSTOMER ENTRY SCREEN"
.
.
.
.
00220 EXIT
***** ***** END OF PROGRAM *****

```

You need to replace the SHOW and WAIT statements (lines 150 and 160, and line 180) with CHAIN statements. (The CHAIN statement is explained in Step 3.) We will do all this in one transaction. Since the cursor is already on line 150, overwrite that line to read:

```
00150 CHAIN"CUST_ENTRY"
```



If you make a mistake when entering the text of a program line, move the cursor to the beginning of the line and press the ERASE EOF key. When you press ENTER, the original line will be returned.



Since we are removing the SHOW statement, we no longer need the associated WAIT statement. Enter "d" over the first zero in line 160, so that it will be deleted when you press ENTER:

```
d0160 ..WAIT
```



---

You can delete a range of lines by entering "DD" on the first and last lines of the range. When you press ENTER, MANTIS deletes all the specified lines.

---

Move the cursor to line 180 and overwrite the line to read:

```
00180 CHAIN"CUST_BROWSE"
```

Now, press ENTER. MANTIS deletes line 160 and makes the changes to lines 150 and 180.



---

Use the RFIND command (PF5 or PF17) if you want to repeat the last FIND command that you issued.

---

### Step 3: Understanding the CHAIN statement

The CHAIN statements that you just added to the menu program (lines 150 and 180) replace the program that is currently executing with the program specified in the CHAIN statement, and begin executing that new program:

```
150 ...CHAIN"CUST_ENTRY"  
...  
180 ...CHAIN"CUST_BROWSE"
```

## Step 4: Renumbering your program

Now, follow these steps to renumber your program:

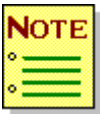
1. Enter the following command on the FSE command line:

```
SEQUENCE 100, 10
```

2. Press ENTER.

MANTIS sequences your program line numbers, beginning at line 100 and incrementing the line numbers by ten. Your program should look like this:

```
00100 ENTRY CUST_MENU
00110 .SCREEN MAP( "CUST_MENU" )
00120 .| THIS IS THE CUSTOMER ACCOUNTS MENU
00130 .CONVERSE MAP
00140 .WHILE MAP<>"CANCEL"
00150 ..WHEN ACTION=1 OR MAP="PF1"
00160 ...CHAIN"CUST_ENTRY"
00170 ..WHEN ACTION=2 OR MAP="PF2"
00180 ...CHAIN"CUST_BROWSE"
00190 ..END
00200 ..CLEAR MAP
00210 ..CONVERSE MAP
00220 .END
00230 EXIT
```



---

Don't run the menu program before replacing it, or your changes will be lost when the CUST\_MENU chains to CUST\_ENTRY or CUST\_BROWSE.

---

## Step 5: Saving your changes

Before running the program, you must save your changes using the REPLACE command. You don't need to specify the program name or password; simply enter REPLACE at the command line and press ENTER. The REPLACE command copies the program in your work area over the existing version in your library.

---

## Exercises

Before proceeding to the next chapter, create the program stubs for the two options on the main Burrys menu. A *stub* is a skeleton program containing temporary code used to execute a simple routine (for example, CONVERSE a screen) and then return to the higher level program.

You will create two stubs, one for the customer entry program (CUST\_ENTRY), and one for the customer browse program (CUST\_BROWSE). Later, you will replace the stubs with actual routines.

Our versions of the program stubs appear in “Exercise examples” on page 237.

### Exercise 1: Creating a stub for the customer entry program

Using the Full-Screen Editor, create a program stub for the customer entry program, naming it CUST\_ENTRY.

This program should display the Burrys new customer screen (CUST\_ENTRY) and prepare it to pass data to the program (hint: refer to your ADD\_CUST program from an earlier exercise). In addition, you must provide a way to exit from the customer entry program and return to the menu program.

All of the statements you need to complete the CUST\_ENTRY stub appear in the menu program.



---

Be sure to save CUST\_ENTRY before you run it.

---

If you design your program stub correctly, you should be able to display the new customer screen and move from it to the menu screen and then back again. When you've finished entering the stub for CUST\_ENTRY, save the program.

## Exercise 2: Creating a stub for the browse program

Create a similar stub for the customer browse screen. Call it CUST\_BROWSE and provide a way to exit from the program stub and return to the menu program.



---

Be sure to save CUST\_BROWSE before you run it.

---

## Exercise 3: Running the menu program

If the CUST\_MENU program is not currently in your work area, enter LOAD"CUST\_MENU" on the command line and press ENTER to load the program into FSE.

Now, run the CUST\_MENU program by entering RUN at the FSE command line and pressing ENTER. When the menu screen displays, enter "1" in the ACTION field and press ENTER.

MANTIS chains to the stub entry program and displays the customer entry screen. Press the CANCEL key to return from this screen to the menu. You can now enter 2 in the ACTION field and press ENTER to chain to the stub browse program and display the customer browse screen.

Press the CANCEL key to return to the menu screen; then, press the CANCEL key again to return from the menu screen to the Full-Screen Editor.

# 7

## Creating a browse program

In this chapter, you'll expand your knowledge of MANTIS programs, and write a browse program that reads records from a file and displays them on a screen. To do so, you will modify the stub for the CUST\_BROWSE program that you created in chapter 6.

### Learning outline

In this chapter, you will:

- ◆ Understand how automatic mapping can reduce the CPU usage and memory requirements of your program.
- ◆ Learn how to specify a MANTIS internal file that your program will access.
- ◆ Learn how to retrieve a record from a MANTIS file.
- ◆ Learn how to access help in the Full-Screen Editor.
- ◆ Learn how to retrieve a series of records from a MANTIS file and display them on a screen.
- ◆ Learn how to use a counter in a MANTIS program.

## Basic concepts: Understanding MANTIS file access

In order to read and write records to a MANTIS file, you must use the FILE statement to identify the file that your program will access:

---

**FILE** *name1*(*[library1:]file-name1,password1*[*,PREFIX*][*,n1*])  
*[,name2*(*[library2:]file-name2,password2*[*,PREFIX*][*,n2*]) . . .]

---

In the FILE statement:

- ◆ **name.** Specifies a symbolic name for the file that you use in subsequent GET, UPDATE, INSERT, and DELETE statements.
- ◆ **[library:] file-name.** Specifies the file name (the name under which the user saved the file during file design.) If the file is located in another user's library, you can access the file by specifying the name of the user library in which it resides (*library:*). If you use the *library* parameter, you must follow it with a colon (:).
- ◆ **password.** Specifies the password that indicates the type of file access your program needs (read only, update, or insert/delete).
- ◆ **PREFIX.** (Optional) Indicates that MANTIS should place the symbolic name and an underscore before all field names defined in this file view.
- ◆ **n.** (Optional) Indicates how many buffers (or LEVELS) MANTIS should allocate to this file.

When you run a program that contains a FILE statement, MANTIS retrieves the file design layout from your library and places it in your work area.

Before you create the browse program, it's important to understand how MANTIS allocates memory for SCREEN and FILE statements. The customer browse screen (CUST\_BROWSE) contains the following fields:

```
CUST_NUMBER  
CUST_NAME  
CUST_BRCH_NUMBER  
CUST_CREDIT_RAT  
CUST_CREDIT_LIM  
CUST_COMMENT  
MESSAGE
```

Each field except MESSAGE occurs 15 times on the screen, due to its vertical repeat specification. When you issue a SCREEN statement, MANTIS allocates memory locations for 15 occurrences of each field. We call these *arrays*.

To distinguish between one occurrence of a field and another, MANTIS adds subscripts from 1 to 15 to each occurrence:

```
CUST_NUMBER( 1 )      CUST_COMMENTS( 1 )  
CUST_NUMBER( 2 )      CUST_COMMENTS( 2 )  
...                   ...  
CUST_NUMBER( 15 )     CUST_COMMENTS( 15 )
```

The Burrys customer file contains elements that have the same names as the browse screen, plus these additional elements:

```
CUST_ADDRESS  
CUST_STATE  
CUST_CLASS  
CUST_CITY  
CUST_ZIP_CODE
```

When you use a FILE statement after a SCREEN statement, MANTIS uses the arrays already allocated by SCREEN, and adds only those variables necessary for symbolic names that are being defined by the file (that is, the five elements listed above).

*Automatic mapping* is the process that MANTIS uses to allow the sharing of data areas between variables of like name and data type. This mapping occurs automatically whenever an already defined name is encountered within a file or screen design. We recommend that you use standard naming conventions on a system-wide basis to make the best use of this feature and avoid moving values from one variable to another.

If you have programs with large numbers of LET statements to move fields between different areas (for example, screens and files), using automatic mapping can reduce the complexity and chance for errors in your program, and also reduce the CPU usage and memory requirements of your program.

With many languages such as COBOL, the programmer defines the memory locations for screen and file variables separately. When data is read, moved, and written, there can be a considerable use of memory and CPU usage. This is not the case with MANTIS. Automatic mapping saves coding time, memory, and CPU usage.

The advantages of using automatic mapping can be lessened if standard naming conventions are not used throughout the shop. Since MANTIS does duplicate definition checking only in certain situations (such as data type), naming conventions should also include standards for text lengths. If these conventions are not followed, it is possible to have a field defined with two different lengths.

For example, it is possible to define a text field as 55 bytes long on a screen, and 35 bytes long in a file. MANTIS uses the first field length encountered when the program is run. That is, if MANTIS encounters the FILE statement first, the data work area is set up with a length of 35 bytes for the field.

Shops that are database-driven should always specify ACCESS, VIEW, TOTAL, and FILE statements first in a MANTIS program to ensure that the MANTIS program always uses the most current definition for a field.



---

## **Step-by-step: Writing a browse program**

In this section, you will create a browse program by filling out the browse stub (CUST\_BROWSE) that you created in chapter 6. The browse program will read the records in the Burrys customer file and display them, 15 at a time, on your browse screen. When MANTIS reaches the end of the file or when you press the CANCEL key, control will return to the menu program.

We'll start with a program that retrieves one record at a time from the Burrys customer file and displays it on the browse screen. To begin, follow these steps to load the CUST\_BROWSE stub:

1. Select the Design a Program option from the Facility Selection menu.
2. From the Program Design Facility menu, enter the EDIT command on the command line (====> EDIT) and press ENTER. The EDIT Program Entry panel displays.
3. Enter the program name CUST\_BROWSE in the From Name field.
4. Leaving the remaining fields blank, press the EXECUTE key (or enter the EXECUTE command on the command line and press ENTER). MANTIS automatically loads and lists the program stub that you created for CUST\_BROWSE.

## Step 1: Specifying a MANTIS internal file

You will use the FILE statement to identify the file that your program will access. Supply a symbolic name, the file name, and the password. At the command line, enter the following line; then, press ENTER:

```
125 FILE REC("CUST_INFO","PASSWORD")
```

When you press ENTER, MANTIS adds line 125 to your program:

```
00100 ENTRY CUST_BROWSE
00110 .SCREEN MAP("CUST_BROWSE")
00120 .CONVERSE MAP
00125 .FILE REC("CUST_INFO","PASSWORD")
00130 .WHILE MAP<>"CANCEL"
00140 ..CONVERSE MAP
00150 .END
00160 .CHAIN"CUST_MENU"
00170 EXIT
```

When you run the preceding program, MANTIS will retrieve the file design layout from your library and place it in your work area.

## Step 2: Retrieving the first record from the file

Next, use the GET statement to retrieve the first record from the customer file. As part of the GET statement, you must supply the file's symbolic name. On the command line, enter the following line and press ENTER:

```
133 GET REC
```

When you press ENTER, MANTIS adds the GET statement to your program:

```
00100 ENTRY CUST_BROWSE
00110 .SCREEN MAP( "CUST_BROWSE" )
00120 .CONVERSE MAP
00125 .FILE REC( "CUST_INFO", "PASSWORD" )
00130 .WHILE MAP<>"CANCEL"
00133 ..GET REC
00140 ..CONVERSE MAP
00150 .END
00160 .CHAIN"CUST_MENU"
00170 EXIT
```



---

When adding lines to a program in the Full-Screen Editor, you can use the INSERT line command to insert blank lines, or you can enter the line (with its line number) at the command line.

---

The GET statement retrieves a record from a file and returns a text value that reflects the status of the GET action. The values are:

Status indicator	Status
FOUND	MANTIS found the record that you requested.
NOTFOUND	The record that you requested does not exist.
NEXT	You issued a GET statement without a key, and MANTIS retrieved the next record in the file.
END	MANTIS failed to retrieve the record because it reached the end of the file.

MANTIS returns these statuses in the file’s symbolic name. In other words, if the GET is successful MANTIS sets the variable REC (your file’s symbolic name) equal to “NEXT”, since you did not supply a key. (You can think of the variable REC as holding a text value that is the status of the prior operation against it.)

Replace and run CUST\_BROWSE. To do so, enter REPLACE and press ENTER; then, enter RUN and press ENTER.

Instead of displaying the customer browse screen, MANTIS returns you to programming mode and displays an error message:

```
NUCIFDE: Internal and file dimensions dissimilar for CUST_NUMBER
```

### Step 3: Accessing help

When you receive an error message, you can use the HELP command to display a prompter. Simply enter HELP at the command line and press ENTER. The following screen displays:

```
NUCIFDE:Internal and file dimensions dissimilar for #####  
  
Explanation:  A FILE statement contains an element whose name is used already by a  
data field in the program.  The dimensions (length of text fields and/or number of  
elements in an array) do not match.  
  
Action:  Locate the element that does not match the previously defined data field and  
make their dimensions match.
```

As the prompter notes, you will usually receive this message when the dimensions for a screen and a file are different. In this instance, your program instructs MANTIS to retrieve one record from the customer file and display it on the screen. But remember, the browse screen has 15 lines on it.

To coordinate your program and screen, you could remove the repeat specifications from the customer browse screen. To do that, you would return to the Screen Design Facility, use the library functions to fetch the screen CUST\_BROWSE, and then use the Update repeat specifications option to change the repeat specifications for each field.

Instead of doing that, we'll expand the CUST\_BROWSE program to display 15 records at a time from the CUST\_INFO file.

## Step 4: Using the LEVEL parameter with the FILE statement

Let's take another look at the FILE statement:

---

```
FILE name1([library1:]file-name1,password1[,PREFIX][,n1])
      [,name2([library2:]file-name2,password2[,PREFIX][,n2]) . . .]
```

---

You need to use the *n1* parameter (that is, LEVEL) in your FILE statement to allocate additional memory for those fields unique to the file.

Since the customer browse screen has 15 lines, you should allocate 15 buffers in your FILE statement. Move the cursor to line 125, and change the line to read.

```
00125 .FILE REC( "CUST_INFO" , "PASSWORD" , 15 )
```

To save your changes, enter REPLACE at the command line and press ENTER.




---

You could also use the PF2 key to replace the program. To view a list of the PF key assignments for various FSE commands, enter PROFILE at the command line and press ENTER.

---

The LEVEL parameter also synchronizes by subscript the elements within a record:

```
CUST_NUMBER( 1 )      CUST_COMMENTS( 1 )
CUST_NUMBER( 2 )      CUST_COMMENTS( 2 )
...                   ...
CUST_NUMBER( 15 )     CUST_COMMENTS( 15 )
```

Thus, each record now contains the correct element at every level, from CUST\_NUMBER through CUST\_COMMENTS.

## Step 5: Clearing the records on the browse screen

The customer browse screen can display up to 15 records at a time. These variables will retain their values until they are set to another value or CLEARED (that is, reset to null or zero). If your customer file contains more than 15 records, you should erase the first 15 records from the screen before you read in the next set.

To do this, add a CLEAR statement after 130. On the command line, enter the following line and press ENTER:

```
131 CLEAR MAP
```

MANTIS adds the line to your program:

```
00100 ENTRY CUST_BROWSE
00110 .SCREEN MAP( "CUST_BROWSE" )
00120 .CONVERSE MAP
00125 .FILE REC( "CUST_INFO" , "PASSWORD" , 15 )
00130 .WHILE MAP<>"CANCEL"
00131 ..CLEAR MAP
```

Now, as the first step in each execution of the WHILE-END loop, MANTIS will erase all prior customer records from the screen.

## Step 6: Adding a counter to the program

MANTIS must keep track of its position as it reads the customer file. To do this, you must use a counter.

The LET statement assigns values to a numeric variable:

---

**[LET]  $\sqrt{\begin{matrix} (i) \\ (i, j) \end{matrix}} \text{ [ROUNDED}(n)] = e1 [,e2,e3 . . .]}$**

---

Give the counter called BUFFER a starting value of 1. To do so, enter the following statement on the command line and press ENTER:

```
132 LET BUFFER=1
```



---

When you set BUFFER equal to 1, you are *initializing* the variable. The LET is also optional; instead, you could just write BUFFER=1. Because the symbolic name BUFFER has not been previously defined, it will become a BIG (numeric) data value.

---

```
00100 ENTRY CUST_BROWSE
00110 .SCREEN MAP( "CUST_BROWSE" )
00120 .CONVERSE MAP
00125 .FILE REC( "CUST_INFO" , "PASSWORD" , 15 )
00130 .WHILE MAP<>"CANCEL"
00131 ..CLEAR MAP
00132 ..LET BUFFER=1
00133 ..GET REC
00140 ..CONVERSE MAP
00150 .END
00160 .CHAIN"CUST_MENU"
00170 EXIT
```



## Step 7: Reading the file record-by-record

Your next step is to construct a second WHILE-END loop that will perform the following tasks:

- ◆ Test for the end of the file or end of the screen occurrences
- ◆ Read the file record-by-record, beginning with the second record
- ◆ Keep track of the buffers on your screen

Examine the following loop:

```
GET REC LEVEL=BUFFER
WHILE REC<>"END"AND BUFFER<15
  BUFFER=BUFFER+1
  GET REC LEVEL=BUFFER
END
CONVERSE MAP
```

The first statement (GET REC LEVEL=BUFFER) tells MANTIS to retrieve a record and place it in the array element(s) represented by BUFFER.

The WHILE statement determines that the loop will continue as long as MANTIS hasn't reached the end of the file. It also prevents the counter from exceeding the number of levels you specified in the FILE statement. When the maximum number of levels has been retrieved from the file, the screen is also full of data.

The next statement (BUFFER=BUFFER+1) increments the counter by 1 each time MANTIS executes the loop. Since you originally set BUFFER equal to one in statement 132, this statement increases it to two the first time that MANTIS executes the WHILE loop. The GET statement retrieves a record from the level (or buffer) equal to the counter.

To modify your program, add four blank lines by entering "i4" on line 133, or the line that contains your GET REC statement. When you press ENTER, four blank lines are inserted after line 133.

Change line 133 and add lines 134 through 137 as shown.

```
00133  ..GET REC LEVEL=BUFFER
        WHILE REC<>"END"AND BUFFERS<15
        BUFFER=BUFFER+1
        GET REC LEVEL=BUFFER
        END
00140  ..CONVERSE MAP
00150  .END
```

## Step 8: Terminating the browse

If you were to run your program now, MANTIS would display the file, 15 records at a time, on your screen. But as the program is currently written you cannot return to the menu, even when you reach the end of the file, without pressing the CANCEL key.

You need to add a condition check that terminates the browse whenever you reach the end of the CUST\_INFO file. Modify line 130 to test for both the CANCEL key and the end of file, as shown:

```
00100 ENTRY CUST_BROWSE
00110  .SCREEN MAP ( "CUST_BROWSE" )
00120  .CONVERSE MAP
00125  .FILE REC( "CUST_INFO" , "PASSWORD" , 15 )
00130  .WHILE MAP<>"CANCEL"AND REC<>"END"
00131  ..CLEAR MAP
00132  ..LET BUFFER=1
00133  ..GET REC LEVEL=BUFFER
00134  ..WHILE REC<>"END"AND BUFFER<15
00135  ...BUFFER=BUFFER+1
00136  ...GET REC LEVEL=BUFFER
00137  ..END
00140  ..CONVERSE MAP
00150  .END
00160  .CHAIN"CUST_MENU"
00170 EXIT
```

This program will display the contents of the entire file, 15 records at a time. Once it has displayed the entire file, MANTIS will automatically return you to the menu screen. But, you can also press the CANCEL key to return to the menu screen before you reach the end of the file. Pressing any key other than CANCEL will display the next set of records.

## Step 9: Sequencing and replacing the browse program

ReNUMBER your program by tens, beginning with 100. To do so, enter the following statement on the command line and press ENTER:

```
SEQUENCE 100,10
```

Now, check your program for accuracy against the following listing:

```
00100 ENTRY CUST_BROWSE
00110 .SCREEN MAP( "CUST_BROWSE" )
00120 .CONVERSE MAP
00130 .FILE REC( "CUST_INFO" , "PASSWORD" , 15 )
00140 .WHILE MAP<>"CANCEL"AND REC<>"END"
00150 ..CLEAR MAP
00160 ..LET BUFFER=1
00170 ..GET REC LEVEL=BUFFER
00180 ..WHILE REC<>"END"AND BUFFER<15
00190 ...BUFFER=BUFFER+1
00200 ...GET REC LEVEL=BUFFER
00210 ..END
00220 ..CONVERSE MAP
00230 .END
00240 .CHAIN"CUST_MENU"
00250 EXIT
```

Before you run your program, be sure to replace it. Enter REPLACE on the command line and press ENTER.

Run your program. You can return to the menu screen by pressing the CANCEL key, no matter how many records the file contains. When you reach the end of the file, MANTIS will automatically return you to the menu screen.

## Exercises

Before proceeding to the next chapter, complete the exercises in this section.

All of the statements you need to complete the exercises appear in CUST\_BROWSE. Our versions of the completed programs appear in “Exercise examples” on page 237.

### Exercise 1: Writing a program to browse the state codes file

Write a browse program that reads the state codes file (STATE\_CODES) and displays one record at a time on the state code entry screen (STATE\_CODE). (Hint: You will not need to use record buffers because only one record at a time will be displayed on the screen.)

The program should read the records in the state codes file until MANTIS reaches the end of the file, or you press the CANCEL key. Save your program, naming it CODE\_BROWSE.

### Exercise 2: Enhancing the state codes browse program

Enhance your state code browse program to display as many state codes as your screen can hold. This means that you must add the necessary repeats for the data field, CUST\_STATE, to the state code screen. (Our screen can hold 14 codes, which means we added 13 repeat specifications.)

Design your program to chain to the menu screen when MANTIS reaches the end of the file, or you press the CANCEL key.

# 8

## Creating a data entry program

Next, you'll use the Full-Screen Editor to expand the customer entry stub (CUST\_ENTRY) that you wrote in chapter 6, to create a data entry program for the Burrys application.

### Learning outline

In this chapter you will learn how to:

- ◆ Understand MANTIS program control capabilities
- ◆ Write a program that accepts your entries and passes control to a subroutine composed of edit checks
- ◆ Display a prompter when the user presses a PF key
- ◆ Write a subroutine that will perform the following tests on the customer information you provide:
  - Verify that the CUST\_NAME field contains an entry
  - Verify the accuracy of the state code that is entered in the CUST\_STATE field
  - Verify that the entry in the CUST\_NUMBER field contains six characters
  - Check CUST\_NUMBER to see if it already exists in the customer file
- ◆ Use the ATTRIBUTE statement to alter the attributes of a data field on an existing screen design

## Basic concepts: Understanding program control basics

This section discusses MANTIS program control capabilities, and explains how to code the DO statement to pass control to a program subroutine.

### Program control capabilities

MANTIS provides the following program control capabilities:

- ◆ **CHAIN.** Allows a MANTIS program to transfer data and control of execution to another MANTIS program without an automatic return path. (You used the CHAIN statement in chapter 6, when you created the CUST\_MENU program.)
- ◆ **DO.** There are two types of DO statements in MANTIS. *Internal* DO allows a MANTIS program to transfer data and control of execution to a routine within the same program, and return to the next statement following the DO. *External* DO allows a MANTIS program to transfer data and control of execution to another MANTIS program with an automatic return path to the next statement. (You will learn how to code both types of DO statements later in this chapter.)
- ◆ **CALL.** Allows a MANTIS program to transfer data and control of execution to a non-MANTIS program and automatically return to the next statement following the CALL. For more information on using the CALL statement, refer to *MANTIS Language, OS/390, VSE/ESA*, P39-5002.
- ◆ **PERFORM.** Allows a MANTIS program to transfer control of execution to a non-MANTIS program and automatically return to the next statement. The PERFORM statement also has options to transfer control without return and to execute background tasks. For more information on using the PERFORM statement, refer to *MANTIS Language, OS/390, VSE/ESA*, P39-5002.

## Using the DO statement

The DO statement transfers program execution to an internal or external subroutine. A *subroutine* is a block of statements either within the existing MANTIS program, or identified by a PROGRAM statement, that performs a function required at one or more points in a program.

Internal DO transfers data and control of execution to a routine *within the same program*, and returns to the next statement following the DO. External DO transfers data and control of execution to *another MANTIS program*, with an automatic return path to the next statement following the DO.

Any program can use both internal and external DO statements. Internal routines are defined by ENTRY statements within the calling program. External routines are defined in PROGRAM statements within the calling program.

External DO allows multiple programs to share common subroutines. Using external DO, you can create programs that can be shared by other users, providing access to both internal and external subroutines. You can also link subroutines with external DO.

Use the following syntax to code the DO statement:

---

**DO entry-name[(argument1,argument2,...)]**

---

In the DO statement:

- ◆ **entry-name.** Specifies the name of a subroutine as indicated in the ENTRY or PROGRAM statement.
- ◆ **argument*n*.** (Optional) Specifies the argument(s) you want passed to the subroutine. A maximum of 255 arguments can be passed in one DO statement.

When you code the DO statement, it must appear on a line by itself. Any additional statements coded on the end of a DO statement (and separated with a colon) are ignored by MANTIS.

You must include an ENTRY-EXIT statement around the subroutine that you specify in the DO statement.

All data names (arguments) must be defined before they are used on a DO statement. Only variables passed as arguments to an external subroutine are available to the external routine. If the variable is a SCREEN, FILE, VIEW, TOTAL, ACCESS, or INTERFACE variable, subvariables are not available unless explicitly passed.

Before you can use the DO statement to invoke an *external* subroutine, you must code the PROGRAM statement to identify the external subroutine to MANTIS.

Use the following syntax to code the PROGRAM statement:

---

**PROGRAM** *name1*(*[library1:]program-name1,password1*)  
*[, name2([library2:]program-name2,password2) . . . ]*

---

In the PROGRAM statement:

- ◆ **namen.** Specifies the symbolic name you will use to refer to your program in subsequent DO statements.
- ◆ **[libraryn:]program-namen.** Specifies the name of the program as you saved it in program design.
- ◆ **password.** Specifies the password as you saved it during program design.



The following program calls both an external subroutine (EDIT\_RTN, lines 40 and 90) and an internal subroutine (ERROR\_RTN, lines 60 and 110):

```

00010 ENTRY EDIT_PROGRAM
00020 .TYPE="CREDIT CHECK"
00030 .PROGRAM EDIT_RTN("VALIDATION", "COMMON")
00040 .DO EDIT_RTN(TYPE,CUST_NO,STATUS,MESSAGE)
00050 .IF STATUS<>"GOOD"
00060 ..DO ERROR_RTN(CUST_NO)
00070 .END
00080 .TYPE="SELECT SALES REP"
00090 .DO EDIT_RTN(TYPE,CUST_NO,STATUS,MESSAGE)
00100 .IF STATUS<>"GOOD"
00110 ..DO ERROR_RTN(SALES_REP)
00120 .ELSE
00130 ..SALES_REP=MESSAGE
00140 .END
00150 EXIT
00160 ENTRY ERROR_RTN(FIELD)
00170 .IF NOTE=" "
00180 ..NOTE=MESSAGE
00190 ..ATTRIBUTE(MAP,FIELD)="BRI,CUR"
00200 .ELSE
00210 ..ATTRIBUTE(MAP,FIELD)="BRI"
00220 .END
00230 EXIT

```

If used optimally, external DO can improve the efficiency of program execution and make it easier to maintain your applications. However, there are many considerations for using external DO effectively that are beyond the scope of the examples provided in this tutorial. For more detailed information on using external DO, refer to *MANTIS Language, OS/390, VSE/ESA*, P39-5002.

# Step-by-step: Creating the customer entry program

You will create the insert module first, and then design your edit checks. To begin, sign on to MANTIS and select the Design a Program option from the Facility Selection menu. On the Program Design Facility command line, enter 1 and press ENTER. The Program Directory List displays:

PRGMLIST01		Program Directory List (BURRYS)		YYYY/MM/DD HH:MM:SS		
====>						
Action	Name	Date	Time	Ver	FMT	Status
-----						
_____	ADD_CUST	2000/08/23	10:16:37	5		ACTIVE
_____	CODE_BROWSE	2000/08/23	10:16:37	5		ACTIVE
_____	CUST_BROWSE	2000/08/23	9:36:27	3		ACTIVE
<b>S</b> _____	CUST_ENTRY	2000/08/23	1:16:40	5		ACTIVE
_____	CUST_MENU	2000/08/23	11:36:59	3		ACTIVE
_____	STATE_CODES	2000/08/23	12:05:42	5		ACTIVE
F1=HELP F2=EXHELP F3=EXIT F4=PROMPT F5=REFRESH F8=FWD F9=RETRIEVE ...						

Enter an s on the blank line next to CUST\_ENTRY; then, press ENTER. MANTIS displays the program stub that you created in chapter 6.

Your completed program must perform these steps:

1. Define the customer information file.
2. Insert the record into the file.
3. Clear your entry from the screen and CONVERSE the screen again.
4. Call the edit subroutine.

After you create the insert module, you will then create a subroutine to edit the data as it is entered.

## Step 1: Defining the file

Begin by adding a FILE statement to the program stub (CUST\_ENTRY) that you developed earlier. Since you'll insert only one record at a time, you don't need to specify a level. On the FSE command line, enter the following statement and press ENTER:

```
115 FILE REC("CUST_INFO","PASSWORD")
```

When you press ENTER, MANTIS adds the line to your program:

```
00100 ENTRY CUST_ENTRY
00110 .SCREEN MAP("CUST_ENTRY")
00115 .FILE REC("CUST_INFO","PASSWORD")
00120 .CONVERSE MAP
00130 .WHILE MAP<>"CANCEL"
00140 ..CONVERSE MAP
00150 .END
00160 .CHAIN"CUST_MENU"
00170 EXIT
```

## Step 2: Inserting a record into the file

The INSERT statement adds a record to a MANTIS file. With INSERT, you must specify the file's symbolic name and, if necessary, the level. Since you are not using a level, your INSERT statement should read:

```
00134 INSERT REC
```

To add this statement, enter it on the command line and press ENTER.

## Step 3: Clearing the entry

After MANTIS inserts the record, you'll want to clear the screen (and thus, all the variables in it) for your next entry (either another customer, or CANCEL). Add the CLEAR statement to your program as shown below:

```
00136 CLEAR MAP
```

Replace and run your program. When the new customer screen appears, you can enter customer information and press ENTER:

```
      B U R R Y S  
NEW CUSTOMER ENTRY
```

```
NAME:  
ADDRESS:  
CITY:  
STATE:  
ZIP CODE:  
  
CUSTOMER NUMBER:  
CLASS:  
CUSTOMER CREDIT RATING:  
CREDIT LIMIT:  
BRANCH NUMBER:  
COMMENTS:
```

## Step 4: Calling the edit subroutine

As it is currently written, your program will insert records into a MANTIS file, but it will not perform any special editing routines on the records you insert. Given the size of the Burrys Corporation, you must provide some basic editing tests to ensure the integrity of the customer file. These edit checks will perform four tests on the data for each new customer that a user enters:

- ◆ Verify that the CUST\_NAME field contains an entry.
- ◆ Verify the accuracy of the state code that is entered in the CUST\_STATE field.
- ◆ Verify that the entry in the CUST\_NUMBER field contains six characters.
- ◆ Check CUST\_NUMBER to see if it already exists in the customer file.

Since the subroutine will check your entries for errors, you should begin by declaring the variable ERROR and setting it equal to FALSE:

```
00131  SMALL  ERROR:ERROR=FALSE
```



FALSE is a MANTIS numeric built-in function that evaluates to 0. Use a numeric variable to hold the Boolean values FALSE (0) and TRUE (1).

Next, you must instruct MANTIS to execute the subroutine. The DO statement transfers program control from a program's current level to a subroutine. Add the following program line and assign a symbolic name to your subroutine in the DO statement as shown:

```
00132 DO VALIDATE_CUST_INFO
```

Now, you need to define the edit subroutine. Add the following statements to your program:

```
00300 ENTRY VALIDATE_CUST_INFO
00310 .SHOW"VALIDATE_CUST_INFO":WAIT
00320 EXIT
```

When MANTIS executes the DO statement in line 132, it will pass control to the edit subroutine in line 300. Then, when MANTIS executes the EXIT statement in line 320, it will return control to the statement following the DO statement (in your program, line 133).

## Step 5: Using the IF-END structure to test for errors

To complete the insert (main line) portion of the program, add an IF-END structure that will insert your record if the subroutine returns a value of FALSE for the variable ERROR. Add these statements to your program as shown:

```
00133 IF NOT(ERROR)
00137 END
```

The NOT function returns TRUE (1) for the expression in line 133 if ERROR evaluates to FALSE (0); otherwise, the NOT function returns FALSE (0).

The IF-END structure executes a block of statements only if the condition you specify is true (in our program, if NOT(ERROR)). Otherwise, MANTIS ignores the block of statements between IF and END. When you include an ELSE statement (IF-ELSE-END) and the condition you specify is false, MANTIS executes the block of statements after ELSE.

Next, add a blank comment on line 180 to make it easier to see the subroutine. Your program should now look like this:

```
00100 ENTRY CUST_ENTRY
00110 .SCREEN MAP("CUST_ENTRY")
00115 .FILE REC("CUST_INFO", "PASSWORD")
00120 .CONVERSE MAP
00130 .WHILE MAP<>"CANCEL"
00131 ..SMALL ERROR:ERROR=FALSE
00132 ..DO VALIDATE_CUST_INFO
00133 ..IF NOT(ERROR)
00134 ...INSERT REC
00136 ...CLEAR MAP
00137 ..END
00140 ..CONVERSE MAP
00150 .END
00160 .CHAIN"CUST_MENU"
00170 EXIT
00180 |
00300 ENTRY VALIDATE_CUST_INFO
00310 .SHOW"VALIDATE_CUST_INFO":WAIT
00320 EXIT
```

## **Step 6: Testing your program structure**

To test your program, temporarily add a vertical bar in front of the CHAIN statement on line 160:

```
00160 |CHAIN"CUST_MENU"
```

Adding the vertical bar changes this line into a comment, so you can run your program without chaining to the CUST\_MENU program. Save your changes by entering REPLACE on the command line and pressing ENTER, then run the program.

When the CUST\_ENTRY screen displays, enter a customer number and press ENTER. MANTIS displays the text string VALIDATE\_CUST\_INFO that follows the SHOW statement (line 310), indicating that MANTIS has successfully passed control to the edit subroutine. Press the CANCEL key twice to return to the Full-Screen Editor.

## Step 7: Adding an edit routine to test for an entry in a field

Your first edit check will test the CUST\_NAME field for an entry. If an operator leaves the text field CUST\_NAME blank, MANTIS sets the field to zero length (NULL). You will use the SIZE function to test for this condition:

---

```
SIZE (field – name [ , "MAX"
                    , "DIM"
                    ,  n
                    , "BYTlength" ] )
```

---

The SIZE function returns the maximum or current length of a field. It can also return the number of defined dimensions for a field or array, as well as the number of occurrences for a specific dimension of an array. In this case, we're interested in seeing if any characters have been entered, so we'll use the default option for returning the current length of the data.

When using this function in your program, you must supply the field name in parentheses:

```
IF SIZE(CUST_NAME)=0
```

If CUST\_NAME equals zero, set the variable, ERROR (initialized at line 131), equal to TRUE:

```
IF SIZE(CUST_NAME)=0
ERROR=TRUE
```




---

TRUE is a MANTIS numeric built-in function that evaluates to +1.

---



Your customer entry screen contains a field named MESSAGE that you can use to return messages to users. If you set MESSAGE equal to a text literal in your program, MANTIS will display the message on the screen:

```
IF SIZE(CUST_NAME)=0
ERROR=TRUE
MESSAGE="CUSTOMER NAME MISSING--PLEASE ENTER"
```




---

Remember to enclose your text literal in double quotes " ".

---

In addition to returning an error message, you can use the ATTRIBUTE statement to highlight the field in error, and place the cursor at the beginning of the field. The ATTRIBUTE statement alters the attributes of a data field on an existing screen design.

After ATTRIBUTE, supply the screen and field names enclosed in parentheses and separated by a comma. To place the cursor in the field, set ATTRIBUTE equal to CUR (an abbreviation for CURSOR). To highlight the field, add BRI (an abbreviation for BRIGHT) after CUR. Enclose them in double quotes, as shown below. For more detailed information on using the ATTRIBUTE statement, refer to *MANTIS Language, OS/390, VSE/ESA*, P39-5002.

Overtyping line 310 with the SIZE function. Next, use the Insert line command to insert 4 blank lines after line 310. (Enter i4 over line number 310 and press ENTER.) Then, add the following statements to your program, so that it looks like this:

```
00310 IF SIZE(CUST_NAME)=0
00311 .ERROR=TRUE
00312 .ATTRIBUTE(MAP,CUST_NAME)="CUR,BRI"
00313 .MESSAGE="CUSTOMER NAME MISSING--PLEASE ENTER"
00314 END
```




---

After you add the statements and press ENTER, press ENTER again to delete the extra blank line.

---

Now, renumber your program to leave room for additional statements. To do so, at the command line enter the following command and press ENTER:

```
SEQUENCE 100,10
```

Your program should look like this:

```
00100 ENTRY CUST_ENTRY
00110 .SCREEN MAP( "CUST_ENTRY" )
00120 .FILE REC( "CUST_INFO" , "PASSWORD" )
00130 .CONVERSE MAP
00140 .WHILE MAP<>"CANCEL"
00150 ..SMALL ERROR:ERROR=FALSE
00160 ..DO VALIDATE_CUST_INFO
00170 ..IF NOT(ERROR)
00180 ...INSERT REC
00190 ...CLEAR MAP
00200 ..END
00210 ..CONVERSE MAP
00220 .END
00230 |CHAIN"CUST_MENU"
00240 EXIT
00250 |
00260 ENTRY VALIDATE_CUST_INFO
00270 .IF SIZE(CUST_NAME)=0
00280 ..ERROR=TRUE
00290 ..ATTRIBUTE(MAP,CUST_NAME)="CUR,BRI"
00300 ..MESSAGE="CUSTOMER NAME MISSING--PLEASE ENTER"
00310 .END
00320 EXIT
```

Remove the comment bar before the CHAIN statement on line 230, then enter the REPLACE command to replace your program. Then, enter the RUN command to run the program.

Press ENTER without keying in a customer name. MANTIS returns your error message at the bottom of the new customer screen, and places the cursor in the first position of the CUST\_NAME field:

```

                                B U R R Y S
                                NEW CUSTOMER ENTRY

NAME:                           —
ADDRESS:
CITY:
STATE:
ZIP CODE:

CUSTOMER NUMBER:
CLASS:
CUSTOMER CREDIT RATING:
CREDIT LIMIT:
BRANCH NUMBER:
COMMENTS:

CUSTOMER NAME MISSING—PLEASE ENTER
```

If you enter a customer name and customer number and press ENTER, MANTIS inserts that record into the Burrys customer file, clears the screen, and waits for your next entry.

Press the CANCEL key to return to the menu screen. Press the CANCEL key again to return to the menu program in the Full-Screen Editor.

At the command line, enter `LOAD CUST_ENTRY`; then, press `ENTER`.  
When you press `ENTER`, MANTIS loads and lists the program:

```

EDIT --- CUST_ENTRY                                COLUMNS 1  73
COMMAND ==>                                         SCROLL ==> PAGE
***** START OF PROGRAM *****
00100 ENTRY CUST_ENTRY
00110 .SCREEN MAP("CUST_ENTRY")
00120 .FILE REC("CUST_INFO","PASSWORD")
00130 .CONVERSE MAP
00140 .WHILE MAP<>"CANCEL"
00150 ..SMALL ERROR:ERROR=FALSE
00160 ..DO VALIDATE_CUST_INFO
00170 ..IF NOT(ERROR)
00180 ...INSERT REC
00190 ...CLEAR MAP
00200 ..END
00210 ..CONVERSE MAP
00220 .END
00230 .CHAIN"CUST_MENU"
00240 EXIT
00250 |
00260 ENTRY VALIDATE_CUST_INFO
00270 .IF SIZE(CUST_NAME)=0
00280 ..ERROR=TRUE
00290 ..ATTRIBUTE(MAP,CUST_NAME)="CUR,BRI"
00300 ..MESSAGE="CUSTOMER NAME MISSING--PLEASE ENTER"

```



Notice that the screen can no longer show the entire program. MANTIS shows only the first 21 lines of the `CUST_ENTRY` program. To view the remainder of the program, use the `DOWN` command (enter `DOWN` on the command line and press `ENTER`) or press `PF8` or `PF20`.

## Step 8: Adding an edit routine to test the number of characters entered

The second edit check also uses the SIZE function, this time to test the CUST\_NUMBER field. The screen will not accept more than six characters, but an operator might inadvertently enter fewer than six characters.

Use the SIZE function to check CUST\_NUMBER for fewer than six characters:

```
IF SIZE(CUST_NUMBER)<6
```

Remember that the SIZE function returns the current size of the field. The < symbol is a MANTIS operator that means "is less than." Otherwise, this test is similar in structure to the first one.

If your IF statement evaluates to true, then MANTIS should set ERROR equal to TRUE and return an appropriate message to the user. Also, MANTIS should place the cursor at the beginning of the field and highlight it:

```
IF SIZE(CUST_NUMBER)<6
ERROR=TRUE
ATTRIBUTE(MAP,CUST_NUMBER)="CUR,BRI"
MESSAGE="CUSTOMER NUMBER MUST BE 6 CHARACTERS"
END
```

Add these statements as part of an ELSE block after your first edit check (that is, the IF block beginning at statement 270):

```
IF expression
    block a (first edit check)
ELSE
    block b (second edit check)
END
```

This means that you must change statement 310 from END to ELSE, add the second IF clause, and add an additional END statement to terminate the original IF clause.

Overtyping line 310 and using the INSERT line command to insert six blank lines after line 310. Then add the new lines and press ENTER so that your subroutine looks like this:

```
00260 ENTRY VALIDATE_CUST_INFO
00270 .IF SIZE(CUST_NAME)=0
00280 ..ERROR=TRUE
00290 ..ATTRIBUTE(MAP,CUST_NAME)="CUR,BRI"
00300 ..MESSAGE="CUSTOMER NAME MISSING--PLEASE ENTER"
00310 .ELSE
00311 ..IF SIZE(CUST_NUMBER)<6
00312 ...ERROR=TRUE
00313 ...ATTRIBUTE(MAP,CUST_NUMBER)="CUR,BRI"
00314 ...MESSAGE="CUSTOMER NUMBER MUST BE 6 CHARACTERS"
00315 ..END
00316 .END
00320 EXIT
```

Notice the program flow in your subroutine. If the SIZE function in statement 270 returns a value of zero, MANTIS executes lines 280-300. Otherwise, MANTIS ignores lines 280-300 and executes line 311.

If the comparison in line 311 returns a value of TRUE, MANTIS executes lines 312-315. If line 311 evaluates to FALSE, MANTIS executes line 320, and control returns to statement 170, IFNOT(ERROR).

After you correct an error on the new customer screen, MANTIS executes the subroutine again from statement 270.

Now, replace and run the program; then, press ENTER without entering a customer name. Again, the message displays asking you to enter a customer name:

```

                                B U R R Y S
                                NEW CUSTOMER ENTRY

NAME:                          —
ADDRESS:
CITY:
STATE:
ZIP CODE:

CUSTOMER NUMBER:
CLASS:
CUSTOMER CREDIT RATING:
CREDIT LIMIT:
BRANCH NUMBER:
COMMENTS:

CUSTOMER NAME MISSING—PLEASE ENTER
```

Enter a customer name and 3 digits of the customer number; then, press ENTER. MANTIS executes the second edit check and highlights CUST\_NUMBER:

```

                                B U R R Y S
                                NEW CUSTOMER ENTRY

NAME:                          Customer Name
ADDRESS:
CITY:
STATE:
ZIP CODE:

CUSTOMER NUMBER:               123
CLASS:
CUSTOMER CREDIT RATING:
CREDIT LIMIT:
BRANCH NUMBER:
COMMENTS:

CUSTOMER NUMBER MUST BE 6 CHARACTERS
```

Notice, however, that both fields remain highlighted. That's because there is nothing in your program to reset the attributes for CUST\_NAME.

To avoid highlighting more than one field at a time, use the ATTRIBUTE statement to RESET field attributes after each execution of the subroutine. Press the CANCEL key to exit from the customer entry screen; then, press the CANCEL key again to exit from the main menu and display the menu program in the Full-Screen Editor.

Load CUST\_ENTRY, then enter the following statement on the command line and press ENTER:

```
215 ATTRIBUTE(MAP) = "RES"
```

This statement tells MANTIS to reset the attributes to their originally defined value (from the SCREEN design) for all the fields on your screen. Use the SEQUENCE command to renumber your program, then replace it and run it again, stepping through the error messages using the procedure that you followed above. This time, when MANTIS returns the second error message, only the CUST\_NUMBER field is highlighted.



## Step 9: Adding an edit routine to validate the state code

The third edit check will test the state code that you enter against the state codes file you set up earlier. If MANTIS doesn't find a record in the state codes file that is identical to your screen entry, then your state code is either missing or incorrect.

You must add two statements to your program before creating the edit routine itself. First, identify and name the state codes file. Supply a unique symbolic name, the library name, and the password:

```
125 FILE REC2("STATE_CODES", "PASSWORD")
```

Secondly, you must tell MANTIS to read the file for a record identical to the entry on your screen. For this, you will use the GET statement with a key:

---

```
GET file-name [(key1, key2, . . . )EQUAL] [ENQUEUE] [LEVEL = n]
                FIRST
                NEXT
                PRIOR
                LAST
```

---




---

A **key** is any file element that is unique to a record. Since a key is unique to a record, you can use it to identify the particular record for which you want MANTIS to read the file.

---

Use the file element, CUST\_STATE, as the key in your GET statement. Supply the file's symbolic name and enclose the key in parentheses:

```
GET REC2(CUST_STATE)
```

If MANTIS returns any status other than "FOUND" for CUST\_STATE, your state code is either missing or incorrect. Thus, the IF statement in your edit check should read:

```
IF REC2<>"FOUND"
```

Otherwise, this edit check is similar to the first two. Add it after line 310, so that MANTIS will edit data fields from the top of the screen to the bottom. Use the INSERT line command to insert six blank lines after line 310, then enter these lines: Here is the additional code you need:

```
00311 GET REC2(CUST_STATE)
00312 IF REC2<>"FOUND"
00313 ERROR=TRUE
00314 ATTRIBUTE(MAP,CUST_STATE)="CUR,BRI"
00315 MESSAGE="INCORRECT OR MISSING CODE"
00316 ELSE
```

You must also add an additional END statement before the EXIT statement in the subroutine.

After you renumber your program, it should look like this:

```
00100 ENTRY CUST_ENTRY
00110 .SCREEN MAP( "CUST_ENTRY" )
00120 .FILE REC( "CUST_INFO" , "PASSWORD" )
00130 .FILE REC2( "STATE_CODES" , "PASSWORD" )
00140 .CONVERSE MAP
00150 .WHILE MAP<>"CANCEL"
00160 ..SMALL ERROR:ERROR=FALSE
00170 ..DO VALIDATE_CUST_INFO
00180 ..IF NOT(ERROR)
00190 ...INSERT REC
00200 ...CLEAR MAP
00210 ..END
00220 ..CONVERSE MAP
00230 ..ATTRIBUTE(MAP)="RES"
00240 .END
00250 .CHAIN"CUST_MENU"
00260 EXIT
00270 |
00280 ENTRY VALIDATE_CUST_INFO
00290 .IF SIZE(CUST_NAME)=0
00300 ..ERROR=TRUE
00310 ..ATTRIBUTE(MAP,CUST_NAME)="CUR,BRI"
00320 ..MESSAGE="CUSTOMER NAME MISSING-PLEASE ENTER"
00330 .ELSE
00340 ..GET REC2(CUST_STATE)
00350 ..IF REC2<>"FOUND"
00360 ...ERROR=TRUE
00370 ...ATTRIBUTE(MAP,CUST_STATE)="CUR,BRI"
00380 ...MESSAGE="INCORRECT OR MISSING CODE"
00390 ..ELSE
00400 ...IF SIZE(CUST_NUMBER)<6
00410 ....ERROR=TRUE
00420 ....ATTRIBUTE(MAP,CUST_NUMBER)="CUR,BRI"
00430 ....MESSAGE="CUSTOMER NUMBER MUST BE 6 CHARACTERS"
00440 ...END
00450 ..END
00460 .END
00470 EXIT
```

Note that this program stops further checking after the first error is found. We could have used a WHEN structure instead of the nested IF statements to find all errors in one pass. But then, you would need some way to determine which error message(s) are displayed when a number of fields are highlighted.



You can highlight all of the fields in error, but only one field gets the initial cursor position on the next CONVERSE. Commonly, the cursor is placed on the top-most field in error. MANTIS will put the cursor on the most recent field receiving the following before the CONVERSE:

```
ATTRIBUTE(map,field)="CUR"
```

An easy way to accomplish this is to edit the fields in reverse sequence and assign CURsor (and error message) whenever you find a field in error. If you do this, the top-most field in error will have the cursor position.

Replace and run your program. Enter a customer name, then enter AA in the CUST\_STATE field and press ENTER. MANTIS returns the error message, but it also overwrites AA with the first record from the state codes file (in our file, AK):

```

      B U R R Y S
    NEW CUSTOMER ENTRY

NAME:                               JACK SMITH
ADDRESS:
CITY:
STATE:                               AK
ZIP CODE:

CUSTOMER NUMBER:
CLASS:
CUSTOMER CREDIT RATING:
CREDIT LIMIT:
BRANCH NUMBER:
COMMENTS:

INCORRECT OR MISSING CODE
```

When MANTIS reads a file, it always returns a record (whether it's the one you requested, or the next record), unless you specified a key in the GET statement or MANTIS has reached the end of the file. In this example, MANTIS couldn't find the invalid AA, so it returned the next higher record in the file (AK).

That record actually appears on your screen because the screen field and the file's key element are both named CUST\_STATE. In other words, when field and element names are identical, MANTIS maps data from the file to the screen. This is another example of *automatic mapping*.

To avoid overwriting data on the screen, you can add the EQUAL parameter to the GET statement. (Another technique used by MANTIS programmers to prevent editing reads from overlaying entered text is to define a second FILE statement with a different symbolic name and use the PREFIX option. This prevents the editing read from automatic mapping to screen fields. Step 11 will present an example of this technique.)

To return directly to the CUST\_ENTRY program, use the TAB key to move the cursor to the lower right corner of the customer entry screen, then enter KILL and press ENTER. KILL will stop an executing program at the CONVERSE statement.

Now, change line 340 as shown:

```
00340 GET REC2(CUST_STATE)EQUAL
```

The EQUAL option tells MANTIS to:

- ◆ Retrieve only a record that has an exact key match to your request.
- ◆ Return a “NOTFOUND” status instead of the “NEXT” record.



---

Use the EQUAL parameter only with a keyed GET.

---

Replace your program and run it. Supply a customer name, and enter AA in the State field. This time, MANTIS returns the error message without retrieving the next record in the state codes file:

```
          B U R R Y S
        NEW CUSTOMER ENTRY

NAME:                JACK SMITH
ADDRESS:
CITY:
STATE:               AA
ZIP CODE:

CUSTOMER NUMBER:
CLASS:
CUSTOMER CREDIT RATING:
CREDIT LIMIT:
BRANCH NUMBER:
COMMENTS:

INCORRECT OR MISSING CODE
```

## Step 10: Displaying the state codes prompter

You can use the PROMPT statement to display informational screens such as the state codes prompter. Supply the library name of the prompter, and enclose it in quotes. Use a WHEN-END structure to show the prompter if the user presses PF1. Enter statements 215-217 as shown:

```
00215 WHEN MAP="PF1"
00216 PROMPT"STATE_CODES"
00217 END
```

Before you run the program, enhance the error message on line 380:

```
00380 MESSAGE="INCORRECT OR MISSING CODE-PRESS 'PF1' FOR HELP"
```



When you enter the previous line, remember to use single quotes (') around PF1 in the message. Alternately, you could use two pairs of double quotes (") around PF1 ("PRESS ""PF1"" FOR HELP").

However, you could *not* use a single pair of double quotes ("PRESS "PF1" FOR HELP") because double quotes designate a text literal and MANTIS would interpret the message as two text literals. If you enter a single pair of double quotes around PF1, MANTIS will generate an error message when you run your program.

Replace and run the program. Then, press PF1 to view the state codes prompter that you built earlier:

STATE CODES		
AL - ALABAMA	KY - KENTUCKY	ND - NORTH DAKOTA
AK - ALASKA	LA - LOUISIANA	OH - OHIO
AZ - ARIZONA	ME - MAINE	OK - OKLAHOMA
AR - ARKANSAS	MD - MARYLAND	OR - OREGON
CA - CALIFORNIA	MA - MASSACHUTSETTS	PA - PENNSYLVANIA
CO - COLORADO	MI - MICHIGAN	RI - RHODE ISLAND
CT - CONNECTICUT	MN - MINNESOTA	SC - SOUTH CAROLINA
DE - DELAWARE	MS - MISSISSIPPI	SD - SOUTH DAKOTA
DC - DISTRICT OF COLUMBIA	MO - MISSOURI	TN - TENNESSEE
FL - FLORIDA	MT - MONTANA	TX - TEXAS
GA - GEORGIA	NE - NEBRASKA	UT - UTAH
HI - HAWAII	NV - NEVADA	VT - VERMONT
ID - IDAHO	NH - NEW HAMPSHIRE	VA - VIRGINIA
IL - ILLINOIS	NJ - NEW JERSEY	WA - WASHINGTON
IN - INDIANA	NM - NEW MEXICO	WV - WEST VIRGINIA
IA - IOWA	NY - NEW YORK	WI - WISCONSIN
KS - KANSAS	NC - NORTH CAROLINA	WY - WYOMING

Now, press ENTER to return to the new customer screen:

B U R R Y S	
NEW CUSTOMER ENTRY	
NAME:	—
ADDRESS:	
CITY:	
STATE:	
ZIP CODE:	
CUSTOMER NUMBER:	
CLASS:	
CUSTOMER CREDIT RATING:	
CREDIT LIMIT:	
BRANCH NUMBER:	
COMMENTS:	
CUSTOMER NAME MISSING—PLEASE ENTER	

Notice that MANTIS returns an error message for the CUST\_NAME field.



Think for a moment about your program's flow:

```

00100 ENTRY CUST_ENTRY
00110 .SCREEN MAP( "CUST_ENTRY" )
00120 .FILE REC( "CUST_INFO" , "PASSWORD" )
00130 .FILE REC2( "STATE_CODES" , "PASSWORD" )
00140 .CONVERSE MAP
00150 .WHILE MAP<>"CANCEL"
00160 ..SMALL ERROR:ERROR=FALSE
00170 ..DO VALIDATE_CUST_INFO
00180 ..IF NOT(ERROR)
00190 ...INSERT REC
00200 ...CLEAR MAP
00210 ..END
00215 ..WHEN MAP="PF1"
00216 ...PROMPT"STATE_CODES"
00217 ..END
00220 ..CONVERSE MAP
00230 ..ATTRIBUTE(MAP)="RES"
00240 .END
00250 .CHAIN"CUST_MENU"
00260 EXIT

```

The first CONVERSE occurs before the while loop (line 140). When you press PF1 at the first CONVERSE, control enters the WHILE loop and executes the validation routine, which issues the error condition because there is no customer name.

Control then passes to the PROMPT statement, which displays the state codes prompt. When you leave the prompt and CONVERSE the screen again, the error message displays.

You can eliminate this problem by testing for PF1 in your program's main module. To do so, enter the following statement on the command line and press ENTER:

```
155 WHEN MAP<>"PF1"
```

Your program should now look like this:

```
00100 ENTRY CUST_ENTRY
00110 .SCREEN MAP( "CUST_ENTRY" )
00120 .FILE REC( "CUST_INFO", "PASSWORD" )
00130 .FILE REC2( "STATE_CODES", "PASSWORD" )
00140 .CONVERSE MAP
00150 .WHILE MAP<>"CANCEL"
00155 .WHEN MAP<>"PF1"
00160 ...SMALL ERROR:ERROR=FALSE
00170 ...DO VALIDATE_CUST_INFO
00180 ...IF NOT(ERROR)
00190 ....INSERT REC
00200 ....CLEAR MAP
00210 ...END
00215 .WHEN MAP="PF1"
00216 ...PROMPT"STATE_CODES"
00217 ..END
00220 ..CONVERSE MAP
00230 ..ATTRIBUTE(MAP)="RES"
00240 .END
00250 .CHAIN"CUST_MENU"
00260 EXIT
```

Replace and run the program again; then, press PF1 to view the prompter. When you return from the prompter to the new customer screen, MANTIS will not display an error message.

## **Step 11: Adding an edit routine to test for a duplicate entry**

The last edit routine tests the customer number to see if it already exists in the Burrys customer file. This edit check is a little more complicated than the first three.

To test for a duplicate customer number, you must tell MANTIS to read the file for a customer number identical to your screen entry. But, if MANTIS finds such a customer number in the file, it retrieves the record and displays it on your screen. In other words, MANTIS will overwrite your screen entry with the record which already contains the customer number you provided.

You can eliminate this problem by using the PREFIX option in your FILE statement. Add the following file statement to your program, supplying a second symbolic name for the Burrys file and specifying PREFIX as shown:

```
00135 .FILE RECX("CUST_INFO", "PASSWORD", PREFIX)
```

PREFIX tells MANTIS to place the symbolic name (in this case, RECX) and an underscore before all field names associated with the file design. Where MANTIS calls the customer number in the unprefixed data area CUST\_NUMBER, it calls the same field in the prefixed data area RECX\_CUST\_NUMBER. In effect, MANTIS sets up a second data area for the Burrys customer file.

This second data area contains independent information as the first data area, and with different names. The second, prefixed, variables can be used in this edit check without fear of overlaying the screen input data with data from the file when the record is read.

If MANTIS doesn't find the customer number in the prefixed file, it is unique and you can enter it into the customer file.

Otherwise, this edit check is similar to the others you've written. Add these statements as an ELSE block after the existing edit checks. This means you must add another END statement before EXIT:

```
00431 ELSE
00432 GET  RECX(CUST_NUMBER)
00433 IF  RECX="FOUND"
00434 ERROR=TRUE
00435 ATTRIBUTE(MAP,CUST_NUMBER)="CUR,BRI"
00436 MESSAGE="DUPLICATE CUSTOMER NUMBER--PLEASE CHANGE"
00437 END
```



---

Use the Insert line command (i7) to add lines 431-437 to your program. (Line 436 is too long to be entered at the command line.)

---

Renumber and replace your program, then compare it with this listing:

```
00100 ENTRY  CUST_ENTRY
00110 .SCREEN MAP("CUST_ENTRY")
00120 .FILE REC("CUST_INFO","PASSWORD")
00130 .FILE REC2("STATE_CODES","PASSWORD")
00140 .FILE RECX("CUST_INFO","PASSWORD",PREFIX)
00150 .CONVERSE MAP
00160 .WHILE MAP<>"CANCEL"
00170 ..WHEN MAP<>"PF1"
00180 ...SMALL ERROR:ERROR=FALSE
00190 ...DO VALIDATE_CUST_INFO
00200 ...IF NOT(ERROR)
00210 ....INSERT REC
00220 ....CLEAR MAP
00230 ...END
00240 ..WHEN MAP="PF1"
00250     PROMPT"STATE_CODES"
```

```
00260 ..END
00270 ..CONVERSE MAP
00280 ..ATTRIBUTE(MAP)="RES"
00290 .END
00300 .CHAIN"CUST_MENU"
00310 EXIT
00320 |
00330 ENTRY VALIDATE_CUST_INFO
00340 .IF SIZE(CUST_NAME)=0
00350 ..ERROR=TRUE
00360 ..ATTRIBUTE(MAP,CUST_NAME)="CUR,BRI"
00370 ..MESSAGE="CUSTOMER NAME MISSING--PLEASE ENTER"
00380 .ELSE
00390 ..GET REC2(CUST_STATE)EQUAL
00400 ..IF REC2<>"FOUND"
00410 ...ERROR=TRUE
00420 ...ATTRIBUTE(MAP,CUST_STATE)="CUR,BRI"
00430 ...MESSAGE="INCORRECT OR MISSING CODE--PRESS 'PF1' FOR HELP"
00440 ..ELSE
00450 ...IF SIZE(CUST_NUMBER)<6
00460 ....ERROR=TRUE
00470 ....ATTRIBUTE(MAP,CUST_NUMBER)="CUR,BRI"
00480 ....MESSAGE="CUSTOMER NUMBER MUST BE 6 CHARACTERS"
00490 ...ELSE
00500 ....GET RECX(CUST_NUMBER)
00510 ....IF RECX="FOUND"
00520 .....ERROR=TRUE
00530 .....ATTRIBUTE(MAP,CUST_NUMBER)="CUR,BRI"
00540 .....MESSAGE="DUPLICATE CUSTOMER NUMBER--PLEASE CHANGE"
00550 ....END
00560 ...END
00570 ..END
00580 .END
00590 EXIT
```



You can scroll your screen by entering DOWN at the command line or by pressing PF8 or PF20. To return to the top of your program, enter TOP at the command line or press PF7 or PF19.

Now, run the program and enter a customer number that already exists in the file:

```

                                B U R R Y S
                                N E W C U S T O M E R   E N T R Y

NAME:                           BOB GREEN
ADDRESS:                        1 MAIN STREET
CITY:                           CINCINNATI
STATE:                           OH
ZIP CODE:                        45000

CUSTOMER NUMBER:                 333333
CLASS:                           10
CUSTOMER CREDIT RATING:         AA
CREDIT LIMIT:                   5000
BRANCH NUMBER:                  1234
COMMENTS:

DUPLICATE CUSTOMER NUMBER--PLEASE CHANGE
```

Next, change the CUST\_NUMBER to one that doesn't already exist in the file and press ENTER. MANTIS inserts the record into CUST\_INFO and clears the error message:

```
B U R R Y S  
NEW CUSTOMER ENTRY
```

```
NAME:  
ADDRESS:  
CITY:  
STATE:  
ZIP CODE:  
  
CUSTOMER NUMBER:  
CLASS:  
CUSTOMER CREDIT RATING:  
CREDIT LIMIT:  
BRANCH NUMBER:  
COMMENTS:
```

Press the CANCEL key to chain to the menu program. Enter 2 in the action field to display the customer browse screen. You'll find the customer you just entered in the file.

## Step 12: Enhancing the customer entry program

When you write programs, flexibility and ease of maintenance are important considerations. For instance, suppose you wanted to change the attributes of fields on your screen when an error condition occurs. To do that, you would have to alter "CUR,BRI" in four places (lines 360, 420, 470, and 530). However, if you replace the text literal "CUR,BRI" with a variable, you can make subsequent changes to the field attributes by changing only a single line in your program.

Use the TEXT statement to name a variable and allocate memory locations for it. With TEXT, you must supply a symbolic name and, optionally, the maximum length (in parentheses) for your variable:

---

```
TEXT name1 [ ([n,] length) ]
           [, name2 ([n,] length) . . . ]
```

---

Enter the following statement on the command line and press ENTER:

```
105 TEXT HIGHLIGHT_ERROR(20)
```

Now the variable HIGHLIGHT\_ERROR is valid for this program, and can hold a text value up to 20 characters long.

Next, replace the text literal, "CUR,BRI" in your subroutine with the variable HIGHLIGHT\_ERROR. Instead of scanning the program for ATTRIBUTE statements, you can issue the CHANGE command:

---

```
{ CHANGE
  CHG
  C } [ , ] [ " ] { fromstring } [ " ] [ , ] [ " ] { tostring } [ " ] [ , ]
                                     [ NEXT
                                     PREV
                                     FIRST
                                     LAST
                                     ALL ] [ , ] [ CHARS
                                     PRE[FIX]
                                     SUF[FIX]
                                     WORD ]
```

---

At the command line, enter the following command and press ENTER:

```
CHG ' "CUR,BRI" ' HIGHLIGHT_ERROR ALL
```

When you press ENTER, MANTIS makes the changes and displays a confirmation message at the top of your screen:

```
4 occurrence(s) of CHARS "CUR,BRI" changed
```



Notice that line 360 is highlighted. This is the first occurrence that was changed as a result of the CHANGE command you issued.



---

If you change a string delineated by double quotes, enclose it in single quotes: ' ' '. If you change a string delineated by single quotes, enclose it in double quotes: " " " ".

---

On the same line where you declared the text variable HIGHLIGHT\_ERROR, you can set it equal to the attributes you specified earlier. Use a colon (:) to separate commands that appear on the same line. Change line 105 as shown:

```
00105 TEXT HIGHLIGHT_ERROR(20):HIGHLIGHT_ERROR="CUR,BRI"
```

Whenever possible, design program modules so that they display on a single screen, so that they're easier to read. In CUST\_ENTRY, neither the main module nor the subroutine meets this criterion. However, there is a way to shorten both routines.

Notice that we initialized ERROR at line 180, and switched it to TRUE four times in the subroutine. Since each edit check also returns a message when it detects an error, we can use the variable MESSAGE to perform the same task as ERROR does now.

Begin by setting MESSAGE equal to null in the subroutine. Enter the following on the command line and press ENTER:

```
00335 MESSAGE=" "
```

Next, make the INSERT at line 200 dependent on the value of MESSAGE, rather than ERROR. Change line 200, as shown:

```
00200 IF MESSAGE=" "
```

Now, MANTIS will insert records into the file only if the MESSAGE field is blank; that is, if there are no errors on your screen. This means that you no longer need the variable, ERROR, or the switches in your program. Therefore, delete statements 180, 350, 410, 460, and 520. Renummer the program again and replace it.

Compare your version with this listing:

```
00100 ENTRY CUST_ENTRY
00110 .TEXT HIGHLIGHT_ERROR(20):HIGHLIGHT_ERROR="CUR,BRI"
00120 .SCREEN MAP("CUST_ENTRY")
00130 .FILE REC("CUST_INFO","PASSWORD")
00140 .FILE REC2("STATE_CODES","PASSWORD")
00150 .FILE RECX("CUST_INFO","PASSWORD",PREFIX)
00160 .CONVERSE MAP
00170 .WHILE MAP<>"CANCEL"
00180 ..WHEN MAP<>"PF1"
00190 ...DO VALIDATE_CUST_INFO
00200 ...IF MESSAGE=""
00210 ....INSERT REC
00220 ....CLEAR MAP
00230 ...END
00240 ..WHEN MAP="PF1"
00250 ...PROMPT"STATE_CODES"
00260 ..END
00270 ..CONVERSE MAP
00280 ..ATTRIBUTE(MAP)="RES"
00290 .END
00300 .CHAIN"CUST_MENU"
00310 EXIT
00320 |
00330 ENTRY VALIDATE_CUST_INFO
00340 .MESSAGE=""
00350 .IF SIZE(CUST_NAME)=0
00360 ..ATTRIBUTE(MAP,CUST_NAME)=HIGHLIGHT_ERROR
00370 ..MESSAGE="CUSTOMER NAME MISSING--PLEASE ENTER"
00380 .ELSE
00390 ..GET REC2(CUST_STATE)EQUAL
00400 ..IF REC2<>"FOUND"
00410 ...ATTRIBUTE(MAP,CUST_STATE)=HIGHLIGHT_ERROR
00420 ...MESSAGE="INCORRECT OR MISSING CODE--PRESS 'PF1' FOR HELP"
```

```
00430 ..ELSE
00440 ...IF SIZE(CUST_NUMBER)<6
00450 ....ATTRIBUTE(MAP,CUST_NUMBER)=HIGHLIGHT_ERROR
00460 ....MESSAGE="CUSTOMER NUMBER MUST BE 6 CHARACTERS"
00470 ...ELSE
00480 ....GET RECX(CUST_NUMBER)
00490 ....IF RECX="FOUND"
00500 ....ATTRIBUTE(MAP,CUST_NUMBER)=HIGHLIGHT_ERROR
00510 ....MESSAGE="DUPLICATE CUSTOMER NUMBER-PLEASE CHANGE"
00520 ....END
00530 ...END
00540 ..END
00550 .END
00560 EXIT
```

Because you have removed four statements from CUST\_ENTRY, you can now view statements 330-530 (including all four edit checks) on a single screen.

## Exercise

Design a maintenance program that will read particular records from the Burrys customer file and delete or update them. Name the program CUST\_MAINT.

Write your DELETE and UPDATE statements in the following formats:

```
DELETE file-name [LEVEL=n]
UPDATE file-name [LEVEL=n]
```

Your program should display the Burrys new customer screen and accept a customer number from the user. If the number exists in the file, MANTIS should display the record and the following message:

```
'PF1' TO DELETE, 'PF2' TO UPDATE, 'PF3' TO CANCEL
```

The program should then perform the following actions:

- ◆ If a user presses PF1, MANTIS should delete the record, clear the screen, and return “DELETION COMPLETE”.
- ◆ If a user presses PF2, MANTIS should update the record, clear the screen, and return “UPDATE COMPLETE”.
- ◆ If a user presses PF3, the program should clear the screen and return “MAINTENANCE CANCELLED AT USER’S REQUEST-- CANCEL TO EXIT”.
- ◆ If MANTIS cannot find a user’s entry, it should clear the screen and return “CUSTOMER NOT FOUND”.

All of the commands you need to write CUST\_MAINT appear in CUST\_BROWSE. Our version of the CUST\_MAINT program appears in “[Exercise examples](#)” on page 237.

# 9

## Using component engineering

This chapter introduces you to the MANTIS Component Engineering Facility (CEF). CEF lets you create MANTIS programs that include reusable subroutines, called components.

Programs created using CEF are known as *component-engineered programs*. You can use CEF to create new component-engineered programs, or to modify existing programs to contain component-engineered code.

Designing and modifying reusable components simplifies your coding effort by allowing the optimal use, reuse, and management of programs. You can modify your components individually, or directly in a composed program.

Using components can also significantly reduce your program testing effort, because a component only needs to be thoroughly tested once, no matter how many applications it is used in.

### Learning outline

In this chapter, you will:

- ◆ Understand basic concepts of component engineering
- ◆ Learn how to define components and decompose a program to create a source program and program components
- ◆ Learn how to use the Compose action to incorporate a component into an existing program
- ◆ Learn how to cross-reference programs
- ◆ Learn how to view the Bill of Materials and Component Where Used lists

## Basic concepts: Understanding component engineering

The Component Engineering Facility (CEF) is a series of options that let you use the methodology of component engineering to design structured applications with reusable components.

A *component* is any set of code that performs a specific function common to more than one program. Components can be subroutines, a group of related subroutines, or just some lines of code such as TEXT, SCREEN, and FILE statements or a program logic fragment.

Components are the “building blocks” of your application because they can be used and reused as necessary at various locations. Once you decide which lines of code will become components in your application, you code COMPONENT statements to include the component code in a source program.

A *source program* is a program that contains MANTIS source code and at least one COMPONENT statement. Source programs are not executable. CEF provides a special command, Compose, that assembles (composes) a source program containing COMPONENT statements and related component code into a composed program that you can edit and run.



---

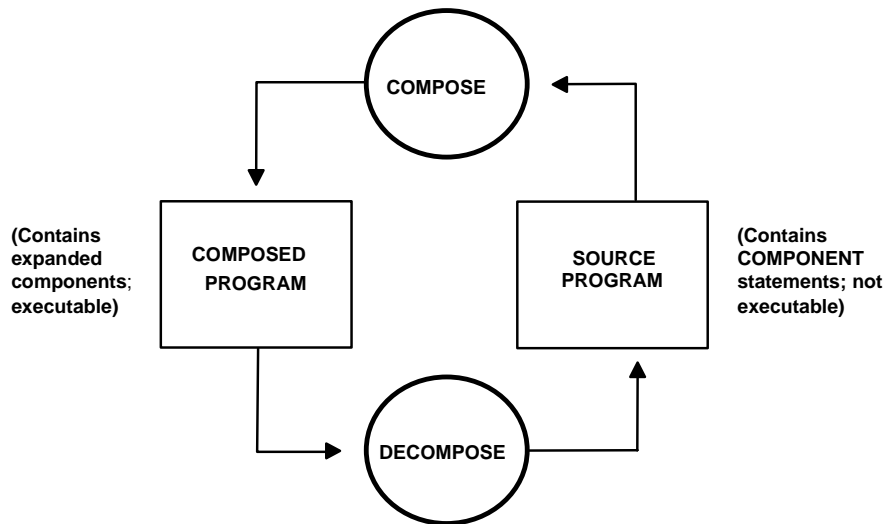
All composed programs are executable programs, but not all executable programs are composed programs. In this tutorial, the term *executable program* refers to any program that can be run in MANTIS. The term *composed program* refers to an executable program that is the result of issuing the Compose action on a source program.

---

When you want to change component code, you can make the change directly in the composed program and issue the Decompose command. Decompose reverses the compose process and disassembles the composed program into source code and component code, automatically updating your library with any changes you have made. Only the source and components you select are replaced in the Decompose process.

In other words, when you change component code you only have to make the change in one component, instead of in multiple programs where the code is used. The Bill of Materials (BILL) process indicates the impact of your change on other programs, by showing you which programs must be Composed to bring in the newly modified component. This entire procedure can be automated.

The following figure provides an overview of the relationship between source programs, the Compose action, composed programs, and the Decompose action:



These processes will be discussed in more detail in subsequent sections.

## Designing components

The first step in designing and coding components is determining what code sections required for your application are good candidates to become components. These code sections will be the ones that you see many times in your system. A component is code that can be used and reused as necessary in various programs throughout the application (or in fact, throughout your entire installation). The components you select may already exist in your application, or you may want to code new ones.

A good example of a component is the standard subroutine used for check-digit computation of account numbers in the banking industry. This subroutine is a typical component because it appears in more than one program of a banking application. A date conversion routine and an initialization routine are two more examples of components.

Like any other MANTIS program, a component is stored in your library. You can store components in a common library, or in separate libraries. If you store components in a common library, it is important to establish naming standards to distinguish between programs and components.

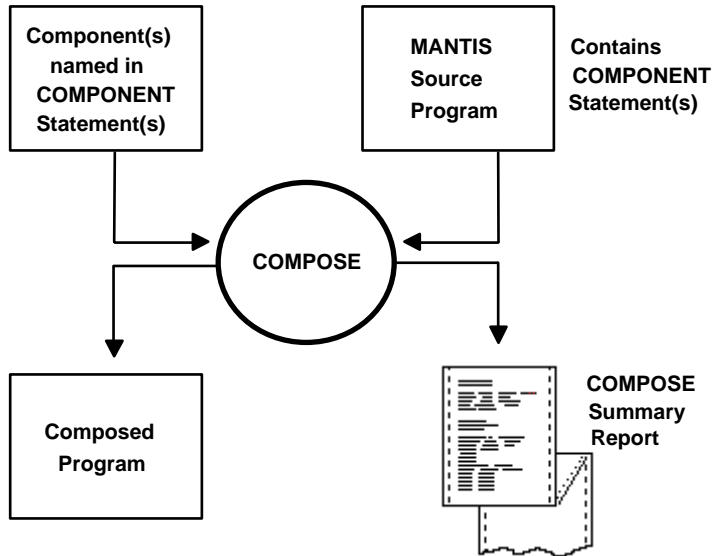
Once components have been created and reside in a program library, you include them in a source program by simply coding a `COMPONENT` statement for each component you want to use. (Code the `COMPONENT` statements where the components are to be brought into the program.)

When you issue the Compose action on the source program, the MANTIS source code and the component code are assembled into a separate, composed program that you can edit and run.



## Creating source programs

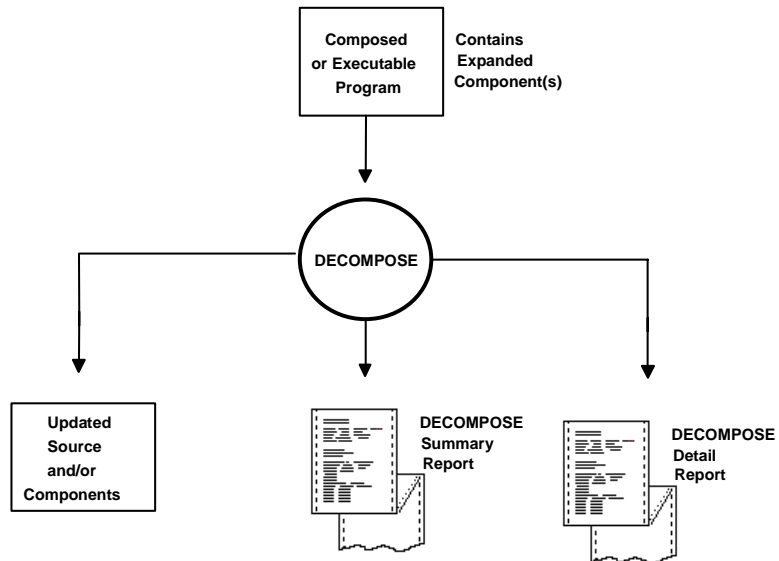
You can create a new source program using the Full-Screen Editor (FSE). A source program is used as input to the Compose action. The following figure shows the starting position of the source program in the Compose action:



You can also modify an existing executable program to contain blocks of code bounded by COMPONENT statements such as this one:

```
| @COMPONENT  
.  
.  
.  
| *CEND
```

You can then issue the Decompose action to split the program into source code and component code:



## Establishing naming conventions for source programs

CEF uses the at sign (@) character as the system default value to identify source programs in your library. The at sign (@) is appended to the end of a source program name. (If you elect to use this naming convention, you will not need to include a REPLACE statement in the source program to identify the name of the composed program.)

When you issue the Compose action to assemble the source program and components into a composed program, CEF will recognize the at sign (@) and automatically assign the composed program with the same name as the source program—minus the at sign (@). For example, if you have a source program named CUST\_BROWSE@, the composed program will be named CUST\_BROWSE unless you override it by specifying some other name in a REPLACE statement.

The following screen illustration shows the Program Directory List displaying source programs and composed programs:

PRGMLIST01		Program Directory List (LIBRARY)			YYYY/MM/DD HH:MM:SS		
====>							
Action	Name		Date	Time	Ver	FMT	Status
-----	-----	-----	-----	-----	---	---	-----
_____	CUST_BROWSE	<----composed	2000/08/23	10:16:37	5	CB	ACTIVE
_____	CUST_BROWSE@	<----source	2000/08/23	9:36:27	3		ACTIVE
_____	CUST_DELETE	<----composed	2000/08/23	1:16:40	5	C	ACTIVE
_____	CUST_DELETE@	<----source	2000/08/23	11:36:59	3		ACTIVE
_____	CUST_INSERT	<----composed	2000/08/23	12:05:42	5	C	ACTIVE
_____	CUST_INSERT@	<----source	2000/08/23	2:10:59	3		ACTIVE
_____	CUST_LIST	<----composed	2000/08/23	4:06:45	5	C	ACTIVE
_____	CUST_LIST@	<----source	2000/08/23	13:37:21	3		ACTIVE
_____	CUST_MAINT	<----composed	2000/08/23	3:51:11	5	CB	ACTIVE
_____	CUST_MAINT@	<----source	2000/08/23	11:33:20	3		ACTIVE
	.						
	.						
	.						
	.						
F1=HELP F2=EXHELP F3=EXIT F4=PROMPT F5=REFRESH F8=FWD F9=RETRIEVE ...							

If you do not use the at sign (@) for source program names, you can use the terms “source”, “composed”, or “executable” in the description text for the program to distinguish between them.

## Accessing component engineering options

You access component engineering options from the Program Design Facility menu. To display this menu, select the Design a Program option from the MANTIS Facility Selection menu. The CEF options are located under the Component Engineering group heading:

```
PRGMMENU01      Program Design Facility (BURRYS)      YYYY/MM/DD HH:MM:SS
===>

Please select one of the menu items below.

      Program      Component Engineering  Bind Options  Utilities
__  1. List        7. CEF Check          12. HPO Check  18. Audit Trail
    2. Edit        8. " Compose          13. " Bind     19. Browse Audit Trail
    3. Profile     9. " Decompose         14. " Unbind   20. " Prgm Profile
    4. Purge       10. CREF Programs       15. SQL Check  21. Trigger List
    5. Copy        11. Bill of Materials    16. " Bind     22. SQL Maint
    6. Rename                        17. " Unbind

FAC000I:READY
F1=HELP  F2=EXHELP  F3=EXIT  F4=PROMPT  F5=REFRESH  F9=RETRIEVE F12=CANCEL ...
```

The following table provides an overview of the component engineering options (underlined text shows the associated commands):

Option name	Description
CEF <u>Check</u>	Identifies program components and source code that changed since the last time you issued the Compose action
CEF <u>Compose</u>	Assembles a source program and its COMPONENT statements into an executable program with expanded component code
CEF <u>Decompose</u>	Disassembles a composed program back into individual components and updates libraries with the latest MANTIS source code and/or component code changes
<u>CREF</u> Programs	Produces a cross-reference (CREF) of the programs or components you designate and then builds the Bill of Materials List from the CREF
<u>Bill</u> of Materials	Displays the Bill of Materials List to show the components referred to in a source program, and displays the Component Where Used List to show the source programs that use a component

---

## Step-by-step: Applying component engineering to the Burrys programs

Now that you have written the CUST\_ENTRY program, you realize that the edit routine will be useful in other programs that you plan to write, so you want to break it out as a component.

At the same time, you want to create a label component that will label the Burrys programs that you have created. You will then incorporate the label component into the all the other Burrys programs (CUST\_BROWSE and CUST\_MENU) so that you can tell at a glance which version of the programs you are editing and running.

To begin, sign on to MANTIS and select the Design a Program option from the Facility Selection menu. On the Program Design Facility command line, enter the following command and press ENTER:

```
EDIT CUST_ENTRY
```



---

When selecting options from the Program Design Facility command line, you can either enter the command associated with the option (for example, EDIT CUST\_ENTRY) or, as a shortcut, enter the option number instead of the command (in this case, 2 CUST\_ENTRY).

---

MANTIS opens the Full-Screen Editor and displays the CUST\_ENTRY program that you created in the previous chapter.



---

The illustrations throughout this tutorial show BURRYS as the user ID and BURRYS as its password. If your own user ID and password are different, use them.

---

## Step 1: Defining the components

You have now loaded the executable program, CUST\_ENTRY, and you want to make the edit routine, VALIDATE\_CUST\_INFO, into a component that can be reused in other programs.

The Decompose action allows you to prototype, develop, refine, and test executable programs. When you finalize these programs, then you can create or update components within the program to make them available for re-use by other source programs.

Components are delineated by a COMPONENT statement (marking the beginning of the component) and a CEND statement (marking the end of the component). The COMPONENT statement is coded as follows:

---

**COMPONENT"[library:] component-name [/password][/description]"**

---

In this case, the COMPONENT and CEND statements will surround the ENTRY and EXIT statements for the subroutine:

```
| *COMPONENT "BURRYS:VALIDATE_CUST_INFO/BURRYS"  
ENTRY VALIDATE_CUST_INFO  
...  
EXIT  
| *CEND
```

When you are decomposing a program, you must nominate (or mark) new and existing updated components in the program with the at sign (@) character. This enables the Decompose action to recognize the components to be decomposed.

Since the COMPONENT and CEND statements don't currently exist in your program, you must add them. At the same time, you will nominate the components for the Decompose action by inserting the at sign (@) in place of the asterisk (\*) shown in the above program lines. The vertical bar (|) must remain as the first character, and the at sign (@) must be the second character in the statement. (You do not have to supply the at sign (@) for the |\*CEND statement.)

Use the FSE command line to add the following program lines:

```
325 | @COMPONENT "BURRYS:VALIDATE_CUST_INFO/BURRYS"  
565 | *CEND
```

You also want to create a new program label component that you will include in all Burrys programs. Creating this component will enable you to update the label in all the Burrys programs by making a single change to the label component.

To create the label component, enter i5 over the first two zeros in line 00100, press ENTER to open five blank lines, then add these lines:

```
| @COMPONENT "BURRYS:BURRYS_PROGRAM_LABEL/BURRYS"
| *****
| BURRYS CUSTOMER ACCOUNTS SYSTEM--MM/DD/YY
| *****
| *CEND
```

Now, renumber your program (SEQUENCE 100,10) and replace it. Your new listing should look like this:

```
00100 ENTRY CUST_ENTRY
00110 . | @COMPONENT "BURRYS:BURRYS_PROGRAM_LABEL/BURRYS"
00120 . | *****
00130 . | BURRYS CUSTOMER ACCOUNTS SYSTEM--MM/DD/YY
00140 . | *****
00150 . | *CEND
00160 .TEXT HIGHLIGHT_ERROR(20):HIGHLIGHT_ERROR="CUR,BRI"
00170 .SCREEN MAP("CUST_ENTRY")
00180 .FILE REC("CUST_INFO","PASSWORD")
00190 .FILE REC2("STATE_CODES","PASSWORD")
00200 .FILE RECX("CUST_INFO","PASSWORD",PREFIX)
00210 .CONVERSE MAP
00220 .WHILE MAP<>"CANCEL"
00230 ..WHEN MAP<>"PF1"
00240 ...DO VALIDATE_CUST_INFO
00250 ...IF MESSAGE=""
00260 ....INSERT REC
00270 ....CLEAR MAP
00280 ...END
00290 ..WHEN MAP="PF1"
00300 ...PROMPT"STATE_CODES"
00310 ..END
```



```
00320 ..CONVERSE MAP
00330 ..ATTRIBUTE(MAP)="RES"
00340 .END
00350 .CHAIN"CUST_MENU"
00360 EXIT
00370 |
00380 |@COMPONENT"BURRYS:VALIDATE_CUST_INFO/BURRYS"
00390 ENTRY VALIDATE_CUST_INFO
00400 .MESSAGE=" "
00410 .IF SIZE(CUST_NAME)=0
00420 ..ATTRIBUTE(MAP,CUST_NAME)=HIGHLIGHT_ERROR
00430 ..MESSAGE="CUSTOMER NAME MISSING--PLEASE ENTER"
00440 .ELSE
00450 ..GET REC2(CUST_STATE)EQUAL
00460 ..IF REC2<>"FOUND"
00470 ...ATTRIBUTE(MAP,CUST_STATE)=HIGHLIGHT_ERROR
00480 ...MESSAGE="INCORRECT OR MISSING CODE--PRESS 'PF1' FOR HELP"
00490 ..ELSE
00500 ...IF SIZE(CUST_NUMBER)<6
00510 ....ATTRIBUTE(MAP,CUST_NUMBER)=HIGHLIGHT_ERROR
00520 ....MESSAGE="CUSTOMER NUMBER MUST BE 6 CHARACTERS"
00530 ...ELSE
00540 ....GET RECX(CUST_NUMBER)
00550 ....IF RECX="FOUND"
00560 ....ATTRIBUTE(MAP,CUST_NUMBER)=HIGHLIGHT_ERROR
00570 ....MESSAGE="DUPLICATE CUSTOMER NUMBER--PLEASE CHANGE"
00580 ....END
00590 ...END
00600 ..END
00610 .END
00620 EXIT
00630 |*CEND
```

Later, when you execute the Decompose action, MANTIS will save the new components in your library.

## Step 2: Coding and nominating the SOURCE statement

Next, you must code a SOURCE statement and nominate it with the at sign (@) so that the Decompose action will recognize this copy of CUST\_ENTRY as the MANTIS source code. (To show how these statements work, you will name the source program CUST\_ENTRY\_SOURCE instead of CUST\_ENTRY@.)

The SOURCE statement names the library (your library only), program, password, and description of the source program to be created or replaced by the Decompose action. The SOURCE statement is coded as follows:

---

```
|@SOURCE"[library:] program-name [/password] [/description]"
```

---

The SOURCE statement must be nominated to be recognized by the Decompose action. Nominate a SOURCE statement by coding the at sign (@) character in the second position following the vertical bar (|), for example |@SOURCE. In addition, be sure you code double quotes (") around the parameters of the SOURCE statement.

To add the SOURCE statement to your program, use the FSE command line to add the following statement:

```
103 |@SOURCE"BURRYS:CUST_ENTRY_SOURCE/BURRYS"
```

### Step 3: Coding the REPLACE statement

Code the REPLACE statement in a source program to identify the program that will be created (or replaced) by the Compose action in the form of a composed program.



---

If you elect to append your source program names with the system default value of the at sign (@), then you do not need a REPLACE statement. However, if there is no REPLACE statement and you do not use the at sign (@), an error message will be displayed when you issue the Compose action.

---

Since you are not using the default value of the at sign (@) to indicate our source program (you are using `_SOURCE` instead), you need to include the REPLACE statement in your program. It is coded as follows:

---

**REPLACE"[*library:*] *program-name* [/password] [/description]"**

---

Double quotes (") are required around the parameters of the REPLACE statement, as shown in the example. To add the REPLACE statement, use the FSE command line to add the following statement:

```
105 | *REPLACE"BURRYS:CUST_ENTRY/BURRYS"
```

Renumber your program (SEQUENCE 100,10) and use the REPLACE command to save your changes to this program. Then, press the CANCEL key to return to the Program Design Facility menu.

## Step 4: Decomposing the program

Now that you have defined your components and coded the SOURCE and REPLACE statements, you are ready to decompose the program. To do so, on the Program Design Facility command line, enter the following line; then, press ENTER:

9 CUST\_ENTRY

MANTIS decomposes the program. When the Decompose action ends, the DECOMPOSE Detail Report displays:

```

PRGMDECO01          DECOMPOSE Detail Report          YYYY/MM/DD HH:MM:SS
                                     More:  -

Library . . . . .      BURRYS
Name . . . . .         CUST_ENTRY
Composed . . . . .
Changed . . . . .      YYYY/MM/DD  HH:MM:SS
Components . . . . .    2

No  Library      Name              Date              Time              D  Stat
--  -
1   BURRYS       BURRYS_PROGRAM_LABEL  YYYY/MM/DD        HH:MM:SS          Y
2   BURRYS       VALIDATE_CUST_INFO    YYYY/MM/DD        HH:MM:SS          Y

CEFCMCI:Components for program
ENTER  F3=EXIT  F12=CANCEL

```

This report lists each component in the executable program, shows the date and time when the component was last replaced in the component library, and indicates individual component status (whether the component has changed since the last time the source program was composed).

When you finish viewing the DECOMPOSE Detail Report, press PF3 or CANCEL. If Function Option "Display summary?" is set to Y (yes) on the Decompose Program Entry panel (the default value), the DECOMPOSE Summary Report displays:

PRGMDECO02	DECOMPOSE Summary Report	YYYY/MM/DD	HH:MM:SS
Decomposed			
Library . . . . .	BURRYS		
Name . . . . .	CUST_ENTRY		
Composed . . . . .			
Changed . . . . .	YYYY/MM/DD	HH:MM:SS	
Source			
Library . . . . .	BURRYS		
Name . . . . .	CUST_ENTRY_SOURCE		
Composed . . . . .			
Changed . . . . .	YYYY/MM/SS	HH:MM:SS	
Options			
	Specified		Default
Comments . . . . .			COMMENTS=YES
Force Compose. . . .			FORCE=NO
Sequence . . . . .			SEQUENCE 10,10
Summary Statistics			
Components . . . . .	2	Replace status . . . . .	AOK
Changed . . . . .		Compose required?. . . . .	YES
Errors . . . . .			
Decompose. . . . .	2		
CEFCRSI: End of summary report			
ENTER F3=EXIT F7=DETAIL F12=CANCEL			

The report highlights the results of the Decompose action, providing the decomposed program information and the source program information.

In addition, the summary statistics include the number of components in the program, the number of components that you changed, any error conditions encountered (such as the number of invalid COMPONENT statements in the program), and the total number of components that were decomposed.

Press PF3 or CANCEL again to return to the Program Design Facility menu.

## Step 5: Viewing the program list

On the Program Design Facility command line, enter 1 and press ENTER. The Program Directory List displays:

```

PRGMLIST01          Program Directory List (BURRYS)          YYYY/MM/DD HH:MM:SS
====>
Action      Name                                           Date      Time      Ver  FMT  Status
-----
      ADD_CUST                                           YYYY/MM/DD HH:MM:SS      2      ACTIVE
      BURRYS_PROGRAM_LABEL                               YYYY/MM/DD HH:MM:SS      2      ACTIVE
      CODE_BROWSE                                         YYYY/MM/DD HH:MM:SS      3      ACTIVE
      CUST_BROWSE                                         YYYY/MM/DD HH:MM:SS      3      ACTIVE
      CUST_ENTRY                                           YYYY/MM/DD HH:MM:SS     12      ACTIVE
      CUST_ENTRY_SOURCE                                   YYYY/MM/DD HH:MM:SS      2      ACTIVE
      CUST_MAINT                                           YYYY/MM/DD HH:MM:SS      3      ACTIVE
      CUST_MENU                                           YYYY/MM/DD HH:MM:SS      4      ACTIVE
      STATE_CODES                                         YYYY/MM/DD HH:MM:SS      2      ACTIVE
      VALIDATE_CUST_INFO                                   YYYY/MM/DD HH:MM:SS      2      ACTIVE

FACF09I:End of file
F1=HELP  F2=EXHELP  F3=EXIT  F4=PROMPT  F5=REFRESH  F8=FWD  F9=RETRIEVE ...

```

Notice that there are now two copies of the customer entry program, the source program CUST\_ENTRY\_SOURCE, and the executable program CUST\_ENTRY.

Also, Decompose has created the component programs `BURRYS_PROGRAM_LABEL` and `VALIDATE_CUST_INFO`. These programs are now available for use as components in other programs.

Press PF3 or CANCEL to return to the Program Design Facility menu.

## Step 6: Adding a component to the customer browse program

Now you are ready to add the new component BURRYS\_PROGRAM\_LABEL to the CUST\_BROWSE program. To do so, you will:

1. Create a new program called CUST\_BROWSE\_SOURCE.
2. Add a SOURCE statement to indicate that this is the source program.
3. Add a REPLACE statement to indicate that the component program should be replaced as CUST\_BROWSE.
4. Code the COMPONENT statement to include the component BURRYS\_PROGRAM\_LABEL in the source program.
5. Save your changes.
6. Use the Compose action to assemble, or compose, the source program and component code into a composed MANTIS program that you can edit and run.

To begin, enter the following line on the Program Design Facility command line and press ENTER:

```
2 CUST_BROWSE
```

MANTIS loads the CUST\_BROWSE program and displays it in the Full-Screen Editor. On the FSE command line, enter the following command and press ENTER:

```
SAVE CUST_BROWSE_SOURCE
```

MANTIS saves a copy of the program CUST\_BROWSE under the new name CUST\_BROWSE\_SOURCE. (If you press ENTER, you will see that you are now editing CUST\_BROWSE\_SOURCE.) Now, enter i3 over the line number 00100 and press ENTER, then enter these statements on the three blank lines:

```
| *SOURCE "BURRYS:CUST_BROWSE_SOURCE/BURRYS"  
REPLACE "BURRYS:CUST_BROWSE/BURRYS"  
COMPONENT "BURRYS:BURRYS_PROGRAM_LABEL/BURRYS"
```

Press ENTER to add the program lines; then, press ENTER again to close the extra blank line.

Note that in a SOURCE program, the compose is not prefixed as a comment (|\* or |@). The COMPONENT in the SOURCE program causes the component to be copied in at that spot. Likewise the REPLACE is not prefixed as a comment in the SOURCE program.

Replace your program and press the CANCEL key to return to the Program Design Facility menu.

On the Program Design Facility command line, enter the following command and press ENTER:

```
8 CUST_BROWSE_SOURCE
```

MANTIS always asks you to confirm the compose action when the executable has changed since the source was last composed. Since this is the first time you've composed this program, the executable has changed since the source was composed, and MANTIS displays the COMPOSE Confirmation panel for you to confirm the compose action:

CONFIRM01

COMPOSE Confirmation

YYYY/MM/DD HH:MM:SS

The executable program shown below has been changed since the last time its source version was composed. Please note that if you want to compose the source program, the changes made to this executable program will be overlaid. Your confirmation is required to continue the Compose action.

Enter Y or N below.

Library	. . . . .	BURRYS
Name	. . . . .	CUST_BROWSE
Error code	. . . . .	CEFCSC
Error message	. . . . .	THE EXECUTABLE PROGRAM HAS CHANGED SINCE THE LAST TIME

Date last changed	. .	YYYY/MM/DD	HH:MM:SS
"	"	composed	. . YYYY/MM/DD HH:MM:SS
"	"	decomposed	. . YYYY/MM/DD HH:MM:SS

Confirmation? . . . . N

Changed by action . . COMPOSE

F1=HELP    F3=EXIT    F12=CANCEL

Enter Y over the N in the Confirmation field and press ENTER.



MANTIS composes your source program and component code into a composed MANTIS program that you can edit and run, and displays the COMPOSE Summary Report:

```

PRGMCOMP02          COMPOSE Summary Report                      YYYY/MM/DD  HH:MM:SS

Source
  Library . . . . .      BURRYS
  Name . . . . .         CUST_BROWSE_SOURCE
  Description . . . . .
Replace
  Library . . . . .      BURRYS
  Name . . . . .         CUST_BROWSE
  Description . . . . .
Options              Specified              Default
Comments . . . . .
Force Compose. . . .
Sequence . . . . .      SEQUENCE 10,10
Summary Statistics
  Replace stmts . . .    1
  Option  " . . .
  Components . . .      1
  Replace status . . .   REP

CSR: END OF SUMMARY REPORT
ENTER  F3=EXIT  F12=CANCEL
  
```

The COMPOSE Summary Report is displayed at the end of the Compose action to highlight the results of the compose. Information on this report includes source and composed program information.

In addition, summary statistics show the number of statements (REPLACE, CSIOPTNS, and COMPONENT) in the source program, as well as the status of the program in your library. (The CSIOPTNS statement is discussed later in this section.)

To end the report, press ENTER. MANTIS returns you to the Program Design Facility menu and displays a confirmation message indicating that the program has been composed.



Once the composed program is created, you can continue to make changes to the source program and issue the Compose action to replace a current composed program as necessary. You can repeat this cycle as often as needed when altering MANTIS source programs or components.

Now that you have composed the new source program CUST\_BROWSE\_SOURCE, take a look at the executable program CUST\_BROWSE. To do so, enter the following on the Program Design Facility command line and press ENTER:

```
2 CUST_BROWSE
```

MANTIS loads the CUST\_BROWSE program and displays it in the Full-Screen Editor:

```
00010 ENTRY CUST_BROWSE
00020 . | *SOURCE "BURRYS:CUST_MENU_BROWSE/BURRYS"
00030 . | *REPLACE "BURRYS:CUST_BROWSE/BURRYS"
00040 . | *COMPONENT "BURRYS:BURRYS_PROGRAM_LABEL/BURRYS"
00050 . | *****
00060 . | BURRYS CUSTOMER ACCOUNTS SYSTEM--MM/DD/YY
00070 . | *****
00080 . | *CEND
00090 .SCREEN MAP ("CUST_BROWSE")
00100 .FILE REC ("CUST_INFO", "PASSWORD", 15)
00110 .WHILE MAP<>"CANCEL"AND REC<>"END"
00120 ..CLEAR MAP
00130 ..LET BUFFER=1
00140 ..GET REC LEVEL=BUFFER
00150 ..WHILE REC<>"END"AND BUFFER<15
00160 ...BUFFER=BUFFER+1
00170 ...GET REC LEVEL=BUFFER
00180 ..END
00190 ..CONVERSE MAP
00200 .END
00210 .CHAIN"CUST_MENU"
00220 EXIT
```

Notice that MANTIS has incorporated the SOURCE and REPLACE statements into CUST\_BROWSE (lines 110 and 120), as well as the component program BURRYS\_LABEL\_PROGRAM (lines 130 through 170).

Notice also that your program line numbers begin at 10 and are incremented by ten. This is the default line number sequence value when MANTIS composes a program. However, if you had wanted your source program line numbers to begin at line 100 instead, you could have specified this sequence value by including a CSIOPTNS statement in your source program.

You can code the CSIOPTNS statement in a source program to specify one or all of three options: COMMENTS, FORCE, and SEQUENCE. COMMENTS, FORCE, and SEQUENCE are keyword parameters that you code in the CSIOPTNS statement (in any order) to specify specific actions when compose is executed. The CSIOPTNS statement is coded as follows:

---

**`[*CSIOPTNS"[COMMENTS=yes] [:FORCE=no] [:SEQUENCE [n,n]"`**

---

The following table lists the CSIOPTNS options and results:

Option	Default value	Result
COMMENTS	Yes	Indicates whether COMPONENT and CEND statements will appear as comments in the composed program
FORCE	No	Indicates whether a source program will be composed (without a warning) if changes were made directly to the composed program since the last time you issued the Compose action. The warning, if requested, will appear as the Compose Confirmation panel when you try to compose the source program
SEQUENCE	(10,10)	Specifies how the composed program lines will be sequenced

So, if you wanted MANTIS to sequence the program beginning at line 100 and incremented by ten, you could have included this line in your source program, directly following the line with the REPLACE statement:

```
CSIOPTNS"SEQUENCE 100,10"
```

You will include the CSIOPTNS statement in a program that you write in the Exercise section for this chapter (see “[Exercise](#)” on page 233. For now, press the CANCEL key to return to the Program Design Facility menu.

### Step 7: Cross-referencing the source program

Next, you need to issue the Cross Reference action (CREF) on the source program. The Cross Reference (CREF) action cross-references programs and components in your library and builds the Bill of Materials List. CREF works by searching source programs for COMPONENT statements, and then displays their relationships and usage in your library when you select the Bill of Materials List.

To cross-reference the source program CUST\_BROWSE\_SOURCE, enter the corresponding option number (10) along with the program name on the Program Design Facility command line and press ENTER:

```
10 CUST_BROWSE_SOURCE
```

MANTIS cross-references the program and, if you set the Function Option “Display summary?” to Y (yes) on the CREF Program Entry panel (the default value), displays the CREF Summary Report:

```
PRGMCREF01          CREF Summary Report          YYYY/MM/DD HH:MM:SS

Source
  Library . . . . . BURRYS
  Name   . . . . . CUST_BROWSE_SOURCE
Statement counts
  COMPONENTS . . . . 1

ENTER  F3=EXIT  F12=CANCEL
```

This report shows the number of components located on your library and cross-referenced by the CREF action. The fields on the report are for viewing only, and cannot be changed. They show that the source program CUST\_BROWSE\_SOURCE contained one COMPONENT statement.

Because CREF builds the Bill of Materials from the cross-reference of source programs and components that it creates, to view the latest Bill of Materials List, you must issue the CREF action when you do the following:

- ◆ After you issue the Copy action to copy the contents of one source program (or component) to another source program. (Issuing CREF includes these copied programs and components in the cross-reference).
- ◆ After you SAVE a new program or component. (Issuing CREF includes new programs and components in the cross-reference).
- ◆ After you issue the Rename action on programs or components. (Issuing CREF includes the renamed programs and components in the cross-reference).
- ◆ After you update (REPLACE) a source program by adding a new component statement. (Issuing CREF includes new program and components in the cross-reference.)

To return to the Program Design Facility menu, press ENTER. Then, at the Program Design Facility command line, enter the following command and press ENTER:

```
10 CUST_ENTRY_SOURCE
```

This command cross-references the program CUST\_ENTRY\_SOURCE. When the CREF Summary Report displays, press ENTER to return to the Program Design Facility menu. Notice that it has two components.

## Step 8: Viewing the Bill of Materials and Component Where Used lists

Now, you are ready to display the Bill of Materials List. Press the PF9 key or enter RETRIEVE on the command line and press ENTER. This retrieves your last command:

10 CUST\_BROWSE\_SOURCE

Overtyping the 10 with an 11; then, pressing ENTER. MANTIS displays the Bill of Materials List for the CUST\_BROWSE\_SOURCE program:

```

EREF0101      BILL OF MATERIALS LIST (BURRYS)      YYYY/MM/DD HH:MM:SS
====>
_____ CUST_BROWSE_SOURCE      PRGM      ACTIVE      BURRYS
Action      Entity Name      Type Relations C Status      Library
-----
_____ CUST_BROWSE      PRGM PRGM SRC      ACTIVE      BURRYS
_____ BURRYS_PROGRAM_LABEL      PRGM COMP SRC      ACTIVE      BURRYS

FACF08I:End of list for this entity
F1=HELP  F2=EXHELP  F3=EXIT  F4=PROMPT  F5=REFRESH  F8=FWD  F9=RETRIEVE ...

```

The components that are contained in CUST\_BROWSE\_SOURCE are listed in the Entity Name field. The Type field displays PRGM (for program), which is the default value.

The Relationship field shows how the items listed under Entity Name relate to the source program shown at the top of the list (CUST\_BROWSE\_SOURCE). The Relationship value of PRGM SRC indicates that the item CUST\_BROWSE is the executable program for CUST\_BROWSE\_SOURCE. The Relationship value of COMP SRC indicates that the item shown is a component (COMP) of the source program.

The field C indicates if any components on this list have changed since the last time the source program was composed. The Status field displays the current program status, and the Library field shows the library in which the components on the list reside.

Some information fields on the Bill of Materials List (Date and Time of last change) extend to the right in columns beyond the width of most screens. If you want to view these fields, issue the RIGHT command at the command line. (To return to the original display of the list, issue LEFT.)

From the Bill of Materials List, you can also display a second list to show the source programs that use a specific component (the Component Where Used List).

To view a list of all the source programs in your library that contain the component BURRYS\_LABEL\_PROGRAM, use the TAB key to move the cursor to the blank line next to BURRYS\_LABEL\_PROGRAM. Enter L (for locate) on the line; then, press ENTER. MANTIS displays the Component Where Used List:

```

EREFLIST01          COMPONENT WHERE USED LIST(BURRYS)          YYYY/MM/DD HH:MM:SS
====>
_____ BURRYS_PROGRAM_LABEL          PRGM          ACTIVE   BURRYS
Action   Entity Name                  Type Relations C Status   Library
-----
_____ CUST_BROWSE_SOURCE             PRGM SRC COMP   ACTIVE   BURRYS
_____ CUST_ENTRY_SOURCE               PRGM SRC COMP   ACTIVE   BURRYS

FACF08I:End of list for this entity
F1=HELP  F2=EXHELP  F3=EXIT  F4=PROMPT  F5=REFRESH  F8=FWD  F9=RETRIEVE ...

```

The Entity Name field lists all the source programs in your library that use the component shown at the top of the list. The Type field defaults to display PRGM (for program).

The Relationship field shows how the source programs listed under Entity Name relate to the component shown at the top of the list, BURRYS\_PROGRAM\_LABEL. The Relationship value of SRC COMP indicates that the items shown on this list are source programs (SRC), and that the entity displayed at the top of the list is a component (COMP) contained in these programs.

The Status field displays the current program status, and the Library field shows the library in which the source programs on the list reside.

Some information fields on the Component Where Used List (Date and Time of last change) extend to the right in columns beyond the width of most terminal screens. If you want to view these fields, issue the RIGHT command at the command line. (To return to the original display of the list, issue LEFT.)

The Bill of Materials List and Component Where Used List are created from the cross-reference of source programs and components generated by the Cross Reference (CREF) action. Each time CREF is issued, a new cross-reference is built for your program. When you display the Bill of Materials List or the Component Where Used List, you are viewing the source program and component cross-reference *as of the last CREF*.

A source program that has no COMPONENT statements will not be shown on the Bill of Materials List or the Component Where Used List.

Press the CANCEL key to return to the Program Design Facility menu.



---

## Exercise

To complete the Burrys programs, add the component `BURRYS_PROGRAM_LABEL` to the third Burrys program, `CUST_MENU`. (Hint: follow the directions in “[Step 6: Adding a component to the customer browse program](#)” on page 223 to add the necessary program statements and compose the program.)

Include the `CSIOPTNS` statement to sequence the program line numbers by ten, beginning at line 100. Our versions of the source and executable programs appear in “[Exercise examples](#)” on page 237.

Once you have created the source program and composed it, cross-reference the new source program, and then view the Bill of Materials for it.



# 10

## What's next?

You've now completed a basic course in MANTIS application development. What's your next step?

First, don't hesitate to go back and redo any of the lessons or exercises, especially those that seemed difficult to you. You'll be surprised how much easier they are the second time through.

Second, you can find more information on MANTIS programming in *MANTIS Program Design and Editing, OS/390, VSE/ESA*, P39-5013. For more detailed information on MANTIS programming statements and functions, refer to *MANTIS Language, OS/390, VSE/ESA*, P39-5002. For more information on the MANTIS design facilities (screen design, file design, etc.), refer to *MANTIS Facilities, OS/390, VSE/ESA*, P39-5001.

Finally, a Reader Comment Sheet appears at the end of this tutorial. If you have any comments on the tutorial, or suggestions for more advanced applications, please take a moment to complete this form and send it to us.



# A

## Exercise examples

This appendix provides our solutions for the exercises in the preceding chapters.

---

### Chapter 1: Introduction

No exercises were presented in chapter 1.

---

### Chapter 2: Creating a screen

In chapter 2, you performed four exercises:

- ◆ Creating the customer accounts menu
- ◆ Creating the new customer entry screen
- ◆ Creating the state code entry screen
- ◆ Viewing the directory of screens

Exercise 1: Creating the customer accounts menu

The customer accounts menu should look like this:

BURRYS

CUSTOMERACCOUNTSYSTEM

ENTERA NEW CUSTOMER | ..... | 1

VIEWCUSTOMERBROWSES SCREEN | ..... | 2

EXITTHIS FACILITY | ..... | CANCEL

: # :

It contains the following field:

Heading	Symbolic name	Field length	Attributes
None	ACTION	1	Numeric

## Exercise 2: Creating the new customer entry screen

The new customer entry screen should look like this:

```

      B|U|R|R|Y|S
    NEW|CUSTOMER|ENTRY

NAME: #####
ADDRESS: #####
CITY: #####
STATE: ##
ZIP|CODE: #####

CUSTOMER|NUMBER: #####
CLASS: ##
CUSTOMER|CREDIT|RATING: #####
CREDIT|LIMIT: #####
BRANCH|NUMBER: #####
COMMENTS: #####

#####
  
```

It contains the following fields:

Heading	Symbolic name	Field length	Attributes
NAME	CUST_NAME	20	Autoskip
ADDRESS	CUST_ADDRESS	20	Autoskip
CITY	CUST_CITY	13	Autoskip
STATE	CUST_STATE	2	Autoskip
ZIP CODE	CUST_ZIP_CODE	5	Numeric, autoskip
CUSTOMER NUMBER	CUST_NUMBER	6	Autoskip
CLASS	CUST_CLASS	2	Autoskip
CUSTOMER CREDIT RATING	CUST_CREDIT_RAT	2	Autoskip
CREDIT LIMIT	CUST_CREDIT_LIM	5	Numeric, autoskip
BRANCH NUMBER	CUST_BRCH_NUMBER	4	Autoskip
COMMENTS	CUST_COMMENTS	25	Autoskip
	MESSAGE	79	Bright, protected

### Exercise 3. Creating the state code entry screen

The state code entry screen should look like this:

STATE | CODE | ENTRY

: ## :

It contains the following field:

Heading	Symbolic name	Field length	Attributes
None	CUST_STATE	2	None



## Exercise 4: Viewing the directory of screens

When you have finished designing and saving the four screens for the Burrys scenario, your Directory of Screens should contain the following screens:

```
DIR003
                                     Directory of Screens                                     YYYY/MM/DD
BURRYS                                                                    HH:MM:SS
-----Name-----  ---Password---  Fmt  -----Description-----
CUST_ENTRY          BURRYS          NEW  BURRYS NEW CUSTOMER ENTRY SCREEN
CUST_MENU           BURRYS          NEW  BURRYS BRANCH MANAGER MENU
CUST_BROWSE         BURRYS          NEW  BURRYS CUSTOMER BROWSE SCREEN
STATE_CODE          BURRYS          NEW  STATE CODE ENTRY SCREEN
```

### Chapter 3: Creating a MANTIS file

The record layout for the STATE\_CODES file should look like this:

MFV003

MANTIS Record Layout Definition

YYYY/MM/DD

Name: STATE\_CODES

HH:MM:SS

Page 1

Element Count 1

Size 32

Element

-----Name-----

Data-type

Dimensions

----Attributes----

1

CUST\_STATE

TEXT

2

KEY

(Use PF1 - PF4 to page; use CANCEL to exit)

### Chapter 4: Creating a prompter

No exercises were presented in chapter 4.

## Chapter 5: Programming fundamentals

In the exercises for this chapter, you:

- ◆ Verified the menu program to ensure that you entered it correctly.
- ◆ Created a program to display a screen and add records to the state codes file.
- ◆ Created a program to display a screen and add records to the customer information file.

### Exercise 1: Verifying the menu program

The menu program you created in chapter 5 should look like this:

```
00100 ENTRY CUST_MENU
00110 .SCREEN MAP ( "CUST_MENU" )
00120 .CONVERSE MAP
00130 .WHILE MAP<>"CANCEL"
00140 ..WHEN ACTION=1 OR MAP="PF1"
00150 ...SHOW"PROGRAM CHAINS TO CUSTOMER ENTRY SCREEN"
00160 ...WAIT
00170 ..WHEN ACTION=2 OR MAP="PF2"
00180 ...SHOW"PROGRAM CHAINS TO CUSTOMER BROWSE SCREEN":WAIT
00190 ..END
00200 .CLEAR MAP
00210 .CONVERSE MAP
00220 .END
00230 EXIT
```

## Exercise 2: Adding records to the state codes file

Your program to add records to the state codes file should look like this:

```
00100 ENTRY STATE_CODES
00110 .SCREEN MAP( "CUST_STATE" )
00120 .FILE REC( "STATE_CODES" , "PASSWORD" )
00130 .CONVERSE MAP
00140 .WHILE MAP<>"CANCEL"
00150 ..INSERT REC
00160 ..CLEAR MAP
00170 ..CONVERSE MAP
00180 .END
00190 EXIT
```

## Exercise 3: Adding records to the customer information file

Your program to add records to the customer information file should look like this:

```
00100 ENTRY ADD_CUST
00110 .SCREEN MAP( "CUST_ENTRY" )
00120 .FILE REC( "CUST_INFO" , "PASSWORD" )
00130 .CONVERSE MAP
00140 .WHILE MAP<>"CANCEL"
00150 ..INSERT REC
00160 ..CLEAR MAP
00170 ..CONVERSE MAP
00180 .END
00190 EXIT
```

---

## Chapter 6: Using the Full-Screen Editor

In chapter 2, you created the stubs for two programs, CUST\_ENTRY and CUST\_BROWSE.

### Exercise 1: Creating a stub for the customer entry program

Your stub for the CUST\_ENTRY program should look like this:

```
00100 ENTRY CUST_ENTRY
00110 .SCREEN MAP( "CUST_ENTRY" )
00120 .CONVERSE MAP
00130 .WHILE MAP<>"CANCEL"
00140 ..CONVERSE MAP
00150 .END
00160 .CHAIN"CUST_MENU"
00170 EXIT
```

### Exercise 2: Creating a stub for the browse program

Your stub for the CUST\_BROWSE program should look like this:

```
00100 ENTRY CUST_BROWSE
00110 .SCREEN MAP( "CUST_BROWSE" )
00120 .CONVERSE MAP
00130 .WHILE MAP<>"CANCEL"
00140 ..CONVERSE MAP
00150 .END
00160 .CHAIN"CUST_MENU"
00170 EXIT
```

## Chapter 7: Creating a browse program

In chapter 7, you created the CODE\_BROWSE program, then enhanced it.

### Exercise 1: Writing a program to browse the state codes file

Your first version of the CODE\_BROWSE program should look like this:

```
00100 ENTRY CODE_BROWSE
00110 .SCREEN MAP( "STATE_CODE" )
00120 .FILE REC( "STATE_CODES" , "PASSWORD" )
00130 .WHILE REC<>"END"AND MAP<>"CANCEL"
00140 ..GET REC
00150 ..CONVERSE MAP
00160 .END
00170 EXIT
```

## Exercise 2: Enhancing the state codes browse program

Your enhanced version of the CODE\_BROWSE program should look like this:

```
00100 ENTRY CODE_BROWSE
00110 .SCREEN MAP( "STATE_CODE" )
00120 .FILE REC( "STATE_CODES" , "PASSWORD" , 14 )
00130 .WHILE REC<>"END"AND MAP<>"CANCEL"
00140 ..CLEAR MAP
00150 ..BUFFER=BUFFER+1
00160 ..GET REC LEVEL=BUFFER
00170 ..WHILE REC<>"END"AND BUFFER<14
00180 ...BUFFER=BUFFER+1
00190 ...GET REC LEVEL=BUFFER
00200 ..END
00210 ..CONVERSE MAP
00220 .END
00230 .CHAIN"CUST_MENU"
00240 EXIT
```

## Chapter 8: Creating a data entry program

In chapter 8, you designed a new maintenance program, CUST\_MAINT. CUST\_MAINT uses many of the same commands and structures as CUST\_BROWSE. The TEXT statement establishes the variable, MSG, and allows it to contain a message up to 60 characters long. The ATTRIBUTE statement locates the cursor at the beginning of the CUST\_NUMBER field.

Within the WHILE-END loop (160-360), MANTIS first tries to retrieve an exact key match for the customer number that a user enters. There are only two possible answers: either the record exists in the file, or it doesn't. Therefore, an IF-ELSE-END structure follows. The IF block consists of WHEN structures that allow the user to choose any one of three options: delete, update, or cancel.

Statements 330-350 clear a user's entry from the screen, set the MESSAGE field equal to the text literal contained in MSG, and prepare the screen for a user's next entry.



The completed CUST\_MAINT program should look like this:

```

00100 ENTRY CUST_MAINT
00110 .SCREEN MAP( "CUST_ENTRY" )
00120 .FILE REC( "CUST_INFO" , "PASSWORD" )
00130 .TEXT MSG(60)
00140 .ATTRIBUTE(MAP,CUST_NUMBER)="CUR"
00150 .CONVERSE MAP
00160 .WHILE MAP<>"CANCEL"
00170 ..GET REC(CUST_NUMBER)EQUAL
00180 ..IF REC="FOUND"
00190 ...MESSAGE=" 'PF1' TO DELETE, 'PF2' TO UPDATE, 'PF3' TO CANCEL"
00200 ...CONVERSE MAP
00210 ...WHEN MAP="PF1"
00220 ....DELETE REC
00230 ....MSG="DELETION COMPLETE"
00240 ...WHEN MAP="PF2"
00250 ....UPDATE REC
00260 ....MSG="UPDATE COMPLETE"
00270 ...WHEN MAP="PF3"
00280 ....MSG="MAINTENANCE CANCELLED AT USER'S REQUEST--CANCEL TO EXIT"
00290 ...END
00300 ..ELSE
00310 ...MSG="CUSTOMER NOT FOUND"
00320 ..END
00330 ..CLEAR MAP
00340 ..MESSAGE=MSG
00350 ..CONVERSE MAP
00360 .END
00370 .CHAIN"CUST_MENU"
00380 EXIT

```

## Chapter 9: Using component engineering

For the exercise in chapter 9, you should have created a new source program called CUST\_MENU\_SOURCE. Your program should look like this one:

```
00100 ENTRY CUST_MENU_SOURCE
00110 .|*SOURCE"BURRYS:CUST_MENU_SOURCE/BURRYS"
00120 .REPLACE"BURRYS:CUST_MENU/BURRYS"
00130 .CSIOPTNS"SEQUENCE 100,10"
00140 .COMPONENT"BURRYS:BURRYS_PROGRAM_LABEL/BURRYS"
00150 .SCREEN MAP( "CUST_MENU" )
00160 .CONVERSE MAP
00170 .WHILE MAP<>"CANCEL"
00180 ..WHEN ACTION=1 OR MAP="PF1"
00190 ...SHOW"PROGRAM CHAINS TO CUSTOMER ENTRY SCREEN"
00200 ...WAIT
00210 ..WHEN ACTION=2 OR MAP="PF2"
00220 ...SHOW"PROGRAM CHAINS TO CUSTOMER BROWSE SCREEN":WAIT
00230 ..END
00240 ..CLEAR MAP
00250 ..CONVERSE MAP
00260 .END
00270 EXIT
```

After you compose the program CUST\_MENU\_SOURCE, if you edit the program CUST\_MENU, it should look like this:

```

00100 ENTRY CUST_MENU
00110 .|*SOURCE"BURRYS:CUST_MENU_SOURCE/BURRYS"
00120 .|*REPLACE"BURRYS:CUST_MENU/BURRYS"
00130 .|*CSIPTNS"SEQUENCE 100,10"
00140 .|*COMPONENT"BURRYS:BURRYS_PROGRAM_LABEL/BURRYS"
00150 .|*****
00160 .| BURRYS CUSTOMER ACCOUNTS SYSTEM--MM/DD/YY
00170 .|*****
00180 .|*CEND
00190 .SCREEN MAP( "CUST_MENU" )
00200 .CONVERSE MAP
00210 .WHILE MAP<>"CANCEL"
00220 ..WHEN ACTION=1 OR MAP="PF1"
00230 ...SHOW"PROGRAM CHAINS TO CUSTOMER ENTRY SCREEN"
00240 ...WAIT
00250 ..WHEN ACTION=2 OR MAP="PF2"
00260 ...SHOW"PROGRAM CHAINS TO CUSTOMER BROWSE SCREEN":WAIT
00270 ..END
00280 ..CLEAR MAP
00290 ..CONVERSE MAP
00300 .END
00310 EXIT

```

---

## Chapter 10: What's next?

No exercises were presented for chapter 10.



# Index

## A

- associated record layout 86
- ATTRIBUTE statement
  - resetting field attributes 184
  - using 177
- automatic mapping
  - defined 151
  - using in programs 152

## B

- bill of materials
  - building 213, 228, 229
  - viewing 230

- blank fill character 34, 38, 42–44, 57, 62–63, 71, 77

## Burrys

- customer
  - entry program, creating 170
  - information file, adding records to 129, 244
- customer accounts system 25–26, 216, 226, 251
- customer browse screen, creating 27, 35
- customer entry screen, creating 79, 237, 239
- customer information file, creating 85
- customer menu
  - program, creating 120
  - screen, creating 77
- programs, applying
  - component engineering to 214
- scenario introduction 24
- state codes
  - file, adding records to 128, 244
  - file, creating xii, 95–96, 128, 164, 185, 188, 190, 243–44, 246
  - prompter, creating 99
  - prompter, displaying 191

**C**

- CANCEL key, defined 23
- CEND statement, using 215
- CHAIN statement, using 145
- CHANGE command, using in the Full-Screen Editor 200
- character set, MANTIS 113
- CLEAR statement, using 137
- column scale line 31–32, 33, 37–38, 65, 67
- component engineering
  - accessing options 212
  - adding a component 223
  - applying to programs 214
  - bill of materials
    - building 213, 228, 229
    - viewing 230
  - component program, creating 214
  - Component Where Used
    - list, viewing the 213, 231
  - composing a program 224
  - decomposing a program 220
  - designing components 208
  - facility, defined 206
  - options 213
  - source programs
    - creating 209
    - cross-referencing 228
    - naming conventions 211
- component program, creating 214
- COMPONENT statement, using 215
- Component Where Used
  - List, viewing 213, 231
- component, defined 206
- component-engineered programs, defined 205
- COMPOSE Summary
  - Report, explanation of 225
- CONVERSE statement, using 134
- CREF Summary Report, explanation 228, 229
- CSIOPTNS statement, using 225, 227

- customer
  - browse screen, creating 27, 35
  - entry
    - program, creating 170
  - entry screen, creating 79, 237, 239
  - information file
    - adding records to 129, 244
  - information file, creating 85
  - menu program, creating 120
  - menu screen, creating 77

**D**

- decimal point character, in programs 112
- DECOMPOSE Summary
  - Report, explanation of 221
- decomposing a program 220
- DEFAULTS command 62
- directory
  - file profiles 94
  - programs 170, 222
  - prompter designs 97, 99, 108
  - screen designs 58, 81
- directory of screens 27, 36, 41, 44, 58, 72, 81, 237, 241
- DO statement
  - internal DO, using 167
  - using 167, 173
- DO statment
  - external DO, using 167
- DOWN command, using in the Full-Screen Editor 180

**E**

- EDIT Program Entry screen
  - using 120, 121
- EDIT Program Entry Screen
  - entry options, specifying 122
  - function options, specifying 122
- edit routine
  - test for a duplicate entry 195
  - test for an entry in a field 176
  - test the number of characters entered 181
  - validate a test string 185
- END command, in the Full-Screen Editor 126
- ENTRY-EXIT statements, using 133
- error message, displaying in a program 179
- expressions
  - defined 114
  - numeric 117
  - text 117–18
- external DO, using 167
- external subroutine, invoking 168

**F**

- Facility Selection menu, standard 22
- field
  - domain 28
- field protected 50
- field specifications
  - listing 53
  - updating 46, 54–55
- file design
  - access levels, defining 86
  - associated record layout 86
  - data type
    - specifying 90
  - directory of files 94
- element
  - attributes 84
  - data type 84
- encrypted format 84, 91
- facility menu 85, 86, 87–88, 92–94, 95–96
- file naming conventions 86
- file profiles, creating or updating 86
- key element 84, 95, 189
- printing 94
- record layout, updating 88
- saving 93
- SCRAMBLE attribute 84, 91
- updating 85

- FILE statement
  - LEVEL parameter 158
  - memory allocation for 151
  - PREFIX option 189
  - using 150, 154
- FIND command, using in the Full-Screen Editor 144
- floating point, numeric data stored in 112
- Full-Screen Editor
  - accessing 120, 122, 131
  - commands
    - defined 132
    - listed 139
    - processing of 138
    - using 138
  - DOWN command, using 180
  - END command 126
  - FIND command, using 144
  - HELP command 157
  - help, accessing in 157
  - immediate mode statements
    - defined 132
    - using 132
  - line commands
    - defined 138
  - LOGOFF command 126
  - primary commands
    - defined 138
  - program
    - adding lines to 155
    - changing a line 200
  - program statements
    - defined 132
    - indentation 132
    - using 133
  - programs, creating in the 123
  - QUIT command 126
  - renumbering programs 146
  - saving program changes 146
  - terminating 120, 126

## G

- GET statement
  - EQUAL parameter 189
  - specifying a key 185
  - status returned 156
- GET statement to retrieve 155

## H

- HELP command, using in the Full-Screen Editor 157
- hidden fields 50

## I

- IF-END structure, using 174
- immediate mode statements
  - defined 132
  - using 132
- initializing variables 160
- internal DO, using 167

## K

- key element 84, 95, 189
- key simulation field 30–31, 65
- KILL command, using 190

## L

- language conventions,
  - MANTIS xii, 112
- LET statement, using 160
- line commands
  - listed 139
- line commands, defined 138
- literals
  - defined 113
  - numeric 117
  - text 117–18
- LOGOFF command 126
- lowercase support 98



**M****MANTIS**

- character set 113
- language conventions xii, 112
- number set 112
- programming commands, using 138
- programming environment xii, 131–32
- signing off 23
- signing on 21, 23
- symbolic names
  - connecting parts of 115
  - using 49, 115

**MANTIS File Design Facility**  
84**MASKED attribute** 53**N**

- naming conventions,
  - standard 134, 151–52
- number set, MANTIS 112

**O**

- opaque map 42, 43–44, 57, 71

**P**

- primary commands
  - listed 139
- Primary commands 138, 140
- primary commands, defined 138
- printing
  - file design 94
  - prompter design 110
  - screen design 59
- Printing the completed
  - screen design 59

**program**

- adding a component 223
- adding lines to 155
- automatic mapping in 152
- changing a line in 200
- comments
  - adding to a program 143
  - defining in programs 137
- component, creating 214
- composing a 224
- control capabilities 166
- control structures 135
- counters, using 160
- decomposing a 220
- directory 170, 222
- error messages, displaying in 179
- memory requirements 149, 152
- program line, finding 144
- renumbering lines in 146
- saving changes to 146
- screen attributes, resetting in 184
- stub, defined 147
- program commands, defined 132

- program design
    - adding a component 223
    - automatic mapping, using 152
    - commands
      - using 138
    - composing a program 224
    - creating programs 123
    - directory of programs 170, 222
    - EDIT Program Entry
      - Screen, using 121
    - EDIT Program Entry
      - Screen, using 120
    - facility menu 111, 120–21, 126, 142, 153, 212, 219, 221–22, 224–25, 227, 229, 232
    - Full-Screen Editor
      - accessing 120, 122, 131
      - commands, listed 139
      - creating programs in 123
      - terminating 120, 126
    - program
      - comments, adding to a 143
      - comments, including 137
      - control capabilities 166
    - program control structures 135
    - program directory 170, 222
    - program line, finding 144
    - program statements, using 133
    - saving program changes 146
    - work area 123
  - PROGRAM statement, using 168
  - program statements
    - defined 132
    - indentation 132
  - program statements, using 133
  - programming environment, MANTIS xii, 131–32
  - PROMPT statement, using 191
  - prompter design
    - chaining to another prompter 98, 107
    - completed design, displaying 97, 109–10
    - creating 103
    - directory 97, 99, 108
    - facility menu 98–99, 100–101, 105–9
    - printing 110
    - saving 106
    - scale line 100, 103
    - tab character, default 97, 100–101
    - tabs, setting 100
    - title 107
    - updating 99, 103
    - work area 100–101, 103–4
  - prototyping, defined 142
- Q**
- QUIT command 126
- R**
- record layout, updating 88
  - repeat specifications
    - listing 55
    - updating 36, 41, 44, 49, 55, 72, 157
  - REPLACE statement, using 219
  - row scale line 29, 31, 33, 37, 56, 65

**S**

- screen attributes, resetting in
  - a program 184
- screen design
  - autoskip attribute 50
  - blank fill character 34, 38, 42–44, 57, 62–63, 71, 77
  - column scale line 31–32, 33, 37–38, 65, 67
  - commands 32
  - completed design, displaying 56
  - data fields 39
  - DEFAULTS command 62
  - directory of screens 58, 81
  - domains 28
  - edit character 40
  - facility menu 36
  - fetching 45
  - field data type 50
  - field specifications
    - listing 53
    - updating 46, 54–55
  - formatting fields 37
  - heading fields 38
  - hidden fields 50
  - key simulation field 30–31, 65
  - library functions 41–42
  - loading into the work area 45
  - mask character 43
  - MASKED attribute 53
  - opaque map 42, 43–44, 57, 71
  - printing 59
  - protected fields 50
  - repeat specifications
    - listing 55
    - updating 36, 41, 44, 49, 55, 72, 157
  - replacing 57
  - reserved fields 31
  - row and column
    - coordinates 30–31, 48, 53, 56, 65, 73, 75
  - saving 41–42
  - selecting the facility 35

- window mode, using 60, 63
  - work area 27, 28–30, 32, 61
- screen domain 28–29, 31, 43, 49, 56, 60–61, 73
- SCREEN statement
  - memory allocation for 151
  - using 134
- SHOW statement, using 136, 142, 144
- Signing off MANTIS 23
- Signing on to MANTIS 21, 23
- SIZE function, using 176, 181
- source program
  - creating 209
  - cross-referencing 228
  - defined 206
  - naming conventions 211
- SOURCE statement, using 218
- standard naming
  - conventions 134, 151–52
- state code entry screen, creating 80, 237, 240
- state codes
  - file
    - adding records to 128, 244
  - file, creating xii, 95–96, 128, 164, 185, 188, 190, 243–44, 246
  - prompter
    - displaying 191
  - prompter, creating 99
- step-wise refinement 142
- subroutine
  - calling an internal 173
  - defined 167
- symbolic name, defined 114
- symbolic names 34, 48–49, 51, 91, 111, 113–16, 151
  - connecting parts of 115
  - examples of 116
  - using 49, 115

## T

tab  
    character, default 97, 100–101  
    settings, prompter design 100  
TEXT statement, using 200

## U

Understanding MANTIS file  
    access 150  
use the ATTRIBUTE  
    statement 165, 177, 184

## V

variables  
    defined 114, 118  
    defining in a program 119  
    initializing 160  
    numeric 117  
test  
    allocating memory for 200  
    text 117–18  
Viewing the directory of  
    screens 58, 81, 237, 241

## W

WHEN statement, using 135, 136, 142, 144  
WHILE-END statements,  
    using 135  
work area  
    program design 123  
    prompter design 100–101, 103–4  
    screen design 27, 28–30, 32, 61